A COMPUTATIONAL APPROACH TO ACHIEVE SITUATIONAL AWARENESS FROM LIMITED OBSERVATIONS OF A COMPLEX SYSTEM

A Thesis
Presented to
The Academic Faculty

by

Jason Sherwin

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
May 2010

COPYRIGHT 2010 BY JASON SHERWIN

A COMPUTATIONAL APPROACH TO ACHIEVE SITUATIONAL AWARENESS FROM LIMITED OBSERVATIONS OF A COMPLEX SYSTEM

Approved by:

Prof. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Brian J. German
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Daniel P. Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. James M. McMichael
Georgia Institute of Technology
Research Institute

Dr. Dileep George
Numenta Inc.

Date Approved: March 29, 2010

To all those who are close to me

ACKNOWLEDGEMENTS

I would like to thank many people. Since this is not an Oscar acceptance speech, I will be able to do so here in a complete fashion. First and foremost, I want to thank my friends and family. My immediate family has been a source of example, guidance and support that I am lucky to have. So I thank my mother, Judith, my father, Byron, and my grandmother, Jean ("Zecie"). As for friends, I do not mean my Facebook friends – although there are many fine chaps and gals bearing that distinction! But I want to thank those souls to whom I believe I am related, though we have not been able to find the familial link as of yet. While completing graduate school, both Martina Stegmeier and Cyril de Tenorio have been such people. They are members of a family none of us have been born into, but to which we all belong. I also must express thanks to my research partner during these years: Thor, thanks so much!

I also want to thank my advisor, Prof. Dimitri Mavris. He has been patient with my development in trying times. He has been supportive of my ideas when the future was uncertain. For these and many other things, I am grateful. Also, I would like to extend my thanks to Profs. Brian German and Daniel Schrage, as well as to Drs. James McMichael and Dileep George. I have valued your guidance throughout the creation of this work.

Finally, I want to acknowledge everyone else that has had an impact, whether it has been joyous or painful. To those who have expressed doubt, here is your proof. To those who have expressed admiration, I hope this does not disappoint. And to those whose paths have not crossed mine, I hope this work gives you insight.

If brevity is the soul of wit then I would be best to stop there. Cue the commercial!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

h	Mass-specific Enthalpy
ρ	Density
u	Uniform Flow Speed
P	Pressure
T	Temperature
V	Vector Space
N	Dimension of Vector Space / Number of Metrics
$C^{i,j}$	Matrix of Coincidence Patterns for j^{th} Node in the i^{th} Level
$G^{i,j}$	Markov-chains of j^{th} Node in the i^{th} Level
g_r	r^{th} Markov-chain
$\lambda_t(g_r)$	Feed-forward Probability Distribution Over Markov-chains at t
$\bar{e}_t$	Bottom-up Evidence at t
HTM	Hierarchical Temporal Memory
SA	Situational Awareness
NCW	Network-Centric Warfare
DoD	Department of Defense
IVPR	Invariant Visual Pattern Recognition

SUMMARY

At the start of the 21st century, the topic of complexity remains a formidable challenge in engineering, science and other aspects of our world. It seems that when disaster strikes it is because some complex and unforeseen interaction causes the unfortunate outcome. Why did the financial system of the world meltdown in 2008-2009? Why are global temperatures on the rise? These questions and other ones like them are difficult to answer because they pertain to contexts that require lengthy descriptions. In other words, these contexts are complex.

But we as human beings are able to observe and recognize this thing we call 'complexity'. Furthermore, we recognize that there are certain elements of a context that form a system of complex interactions – i.e., a complex system. Many researchers have even noted similarities between seemingly disparate complex systems. Do sub-atomic systems bear resemblance to weather patterns? Or do human-based economic systems bear resemblance to macroscopic flows? Where do we draw the line in their resemblance? These are the kinds of questions that are asked in complex systems research.

And the ability to recognize complexity is not only limited to analytic research. Rather, there are many known examples of humans who, not only observe and recognize but also, operate complex systems. How do they do it? Is there something superhuman about these people or is there something common to human anatomy that makes it possible to fly a plane? – Or to drive a bus? Or to operate a nuclear power plant? Or to play Chopin's etudes on the piano? In each of these examples, a human being operates a

complex system of machinery, whether it is a plane, a bus, a nuclear power plant or a piano. What is the common thread running through these abilities?

The study of situational awareness (SA) examines how people do these types of remarkable feats. It is not a bottom-up science though because it relies on finding general principles running through a host of varied human activities. Nevertheless, since it is not constrained by computational details, the study of situational awareness provides a unique opportunity to approach complex tasks of operation from an analytical perspective. In other words, with SA, we get to see how humans observe, recognize and react to complex systems on which they exert some control.

Reconciling this perspective on complexity with complex systems research, it might be possible to further our understanding of complex phenomena if we can probe the anatomical mechanisms by which we, as humans, do it naturally. At this unique intersection of two disciplines, a hybrid approach is needed. So in this work, we propose just such an approach.

In particular, this research proposes a computational approach to the situational awareness (SA) of complex systems. Here we propose to implement certain aspects of situational awareness via a biologically-inspired machine-learning technique called Hierarchical Temporal Memory (HTM). In doing so, we will use either simulated or actual data to create and to test computational implementations of situational awareness. This will be tested in two example contexts, one being more complex than the other. The ultimate goal of this research is to demonstrate a possible approach to analyzing and understanding complex systems. By using HTM and carefully developing techniques to analyze the SA formed from data, it is believed that this goal can be obtained.

CHAPTER 1

MOTIVATION

The title of this work is "A computational approach to achieve situational awareness from limited observations of a complex system." There are many disciplines whose efforts have led to the approach set forth in this work. From complex systems to human factors, and from network-centric warfare to machine learning, there is a wealth of literature that has identified salient points that motivate this research. In the following sections, this literature will be reviewed. The goal of this literature survey is to identify a common thread running through these disciplines. This common thread is not obvious; however, once it is identified, its importance for engineering application will become apparent. In particular, it will be shown how a computational approach with its roots in various aspects of biology can be used to comprehend a complex system.

Complex Systems

The literature on complex systems is as broad as the topic on which it is focused. However, there are key results from the cutting edge of this discipline that can serve as a guide. In particular, the following sections will consider work from the Santa Fe Institute and from other specialists in the dynamics of complex systems. As will be shown in the course of this survey, it is not easy to separate the study of complex systems from 'who' or 'what' is doing the studying. This relationship manifests itself in determining how a particular complex system is observed, modeled and ultimately understood.

Into the Heart of Complex Systems: Work from the Santa Fe Institute

Murray Gell-Mann's book *The Quark and the Jaguar* [1] provides an excellent narrative of what complex systems research is all about. He begins by highlighting the importance of information in its study: "In studying any complex adaptive system, we

follow what happens to the information. We examine how it reaches the system in the form of a stream of data ... [and how it condenses it] into a schema [.]" Notice here that the term 'complex *adaptive* system' is used, rather than just 'complex system'. This distinction emphasizes the point that a complex system can react, or *adapt*, to changes within its components' behaviors systemically.

Such adaptation is a central feature to a complex system. Furthermore, this ability to adapt to changes in component behavior is what makes the examination of the data stream so important. This stream of data, or "information," is all we know about how the complex system behaves. As Yaneer Bar Yam says in his book, *Dynamics of Complex Systems*, "there is only one property of the complex system that we know for sure – that it is complex" [2]. Consequently, it is the task of anyone studying a complex system to follow its information in order to unravel its complexity. Indeed, this is a monumental task when faced with the slew of information that describes a complex system. It is therefore necessary to consider the schema into which groups of data are organized.

The idea of schema in complex systems will be important to the discussion of situational awareness. A schema can be thought of as a way to condense information. Schemata are extremely important in complex systems research because they are the prime mechanism by which complexity is removed from the system's accurate description. Consequently, in a field of research that seeks the regularities and shortest descriptions of a complex system, the formation and evolution of a schema is of utmost importance. In his summary of how a complex system works, Gell-Mann shows the interaction between information (or *data*) and the schema used to compress that information (Figure 1). We can see from this figure that data is what contributes to the formation of a schema. In particular, at some time instant there has been an accumulation of "[previous] data, including behaviors and effects" that has led to the current "schema that summarizes and is capable of predicting." There is an important intermediate step between the raw data and the schema. Specifically, "regularities" in the information are

identified to enable "compression." This is how a schema is made from previous information.

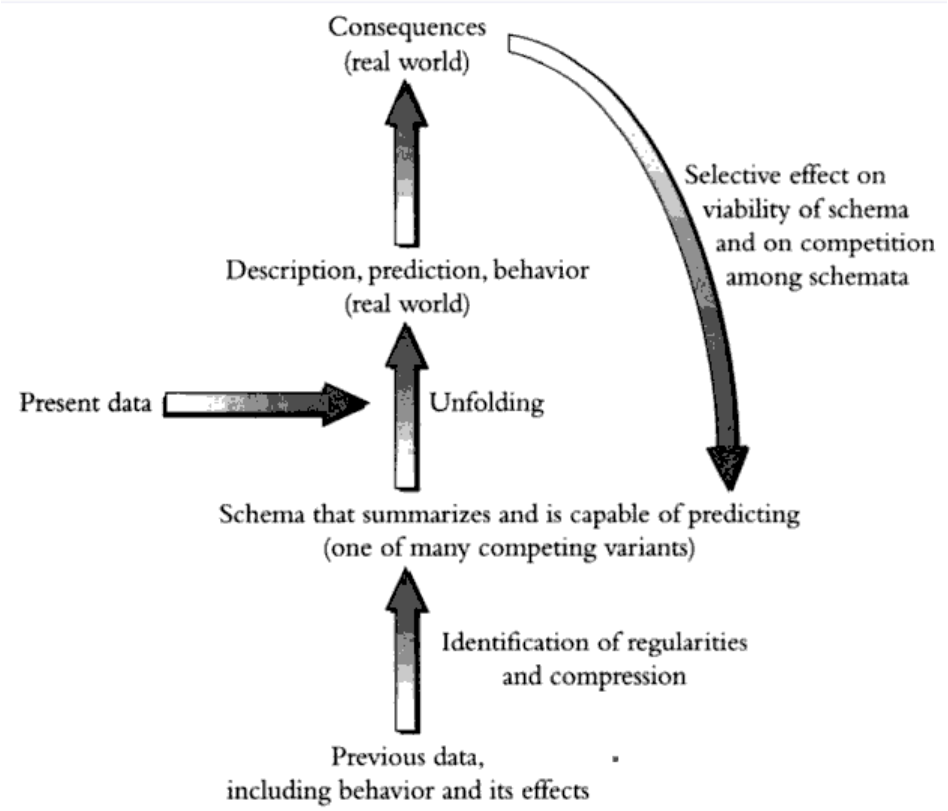


Figure 1: How a Complex System Works [1]

But a schema is not formed at one instant and never changed. In fact, the dynamic nature of a complex system is what makes its short description so difficult. Consequently, it necessitates that its schema evolves if it is to remain an effective tool of summary and prediction. This is why "present data" is observed from which "description" and "consequences" exert a "selective effect on the viability of [a] schema." So not only the attainment but also the maintenance of a schema is the central focus of complex systems research. As Figure 1 shows, this is a dynamic process.

Another key aspect to complex systems research is the issue of *coarse graining*. Coarse graining is a specification of the "level of detail up to which the system is described, with finer details being ignored," according to Gell-Mann. This makes sense

as something worth considering because data plays a central role in the formation of a schema. So it is necessarily important to specify the level of detail in data because the schema will necessarily be affected by it.

This issue of coarse graining in fact opens the door to another key aspect of complex systems research that has thus far been implied: *context dependence*. As Gell-Mann puts it, context dependence "keeps cropping up in attempts to define different kinds of complexity." This is because the level of coarse graining has to do with the ways by which information is extracted from the complex system. So the question of 'who' or 'what' is forming a schema becomes an important consideration. On this point, Gell-Mann's view differs from that of Bar Yam. Specifically, Gell-Mann insists that "complexity is defined in terms of the length of a description" and so "it is not an intrinsic property of the thing described." He continues to write that this is true because "the length of a description may depend on who or what is doing the describing." This perspective differs from Bar Yam's from earlier because, according to Gell-Mann, complexity is not the only "property of the complex system that we know for sure." Rather, Gell-Mann asserts that this property (i.e., complexity) is not known unless the length of description is suitably long, and this depends on the "what" or the "who" describing the complex system. Though neither of these perspectives is wrong, Gell-Mann highlights the importance of specifying the what, who or *other system* that describes a complex system. Once this context has been specified, the level of coarse graining follows.

Since context is important to schema formation, it is useful to consider some possible contexts by which a schema can be formed. As Gell-Mann puts it, "we are discussing one or more definitions of complexity that depend on a description of one system by another system, presumably a complex adaptive system, which could be a human observer." As the distinction between Bar Yam's notion of complexity and that of Gell-Mann's has shown, the root of the distinction deals with the description of one

system by another. Specifically, we see from this argument that complex systems research is somewhat of a "fight-fire-with-fire" discipline. For instance, in many cases of expertise, a human observer is the system describing a complex system. But Gell-Mann points out that "[computers] can function as complex adaptive systems ... computers with ordinary hardware can be programmed to learn or adapt or evolve." Complex systems research therefore is not something relegated to human-based description or understanding. In fact, a computing machine can form its own schema about another complex system it observes. As Gell-Mann points out, "most such designs or programs have depended on imitating a simplified picture of how some living complex adaptive system works." And, this is precisely why complex systems are related to both situational awareness and computation. This overlap will become more apparent as we proceed.

Technical Details to Studying Complex Systems

Murray Gell-Mann's book, *The Quark and the Jaguar*, initiated the discussion on complex systems research quite well. The important issues of information, schema, coarse graining and context dependence have opened the door to what will be the focus of this research. However, there are further levels of detail to encounter when each of these issues are probed in greater detail. Along the way, we wish to know what techniques have been developed to deal with each of these issues.

Let us start with context dependence, which is important for determining the length of the description used to characterize a complex system. But the "length of the description" is a subjective notion thus far. Bar Yam points out that in complex systems research "[the] central issue is defining quantitatively what we mean by complexity." How long is a 'long description'? Can one complex system have a shorter description than another? And then does that shorter description qualify that system as being less complex? With these questions in mind, it is important to develop a quantitative understanding of complexity. Bar Yam points out that to do so "we will use tools of both

statistical physics and computer science – information theory and computation theory. According to this understanding, complexity is the amount of information necessary to describe a system." Of course, this is a similar notion of complexity to that of Gell-Mann. In his words, the length of the description is what Bar Yam calls the "amount of information necessary to describe." But as Gell-Mann points out, both the level of detail (i.e., the coarse graining) and the context (i.e., the person or system doing the describing) are important to specify beforehand.

So given a coarse graining and a context, we can begin to quantify complexity. To quantitatively define complexity, some tools from established disciplines are of use. However, it is necessary to know the bounds of their applicability. For instance, statistical physics and computer science have been mentioned thus far. In fact, "[all] approaches that are used for the study of simple systems can be applied to the study of complex systems. However, it is important to recognize features of conventional approaches that may hamper progress in the study of complex systems [.]" Bar Yam continues to point out three caveats when studying complex systems:

- "Don't take it apart. Since interactions between parts of a complex system are essential to understanding its behavior, looking at parts by themselves is not sufficient. It is necessary to look at parts in the context of the whole...
- Don't assume smoothness...
- Don't assume that only a few parameters are important [.]"

These caveats address the hallmarks of mathematical descriptions employed for simple systems. Simple systems are "composed of complex parts where the collective behavior is simple," as Bar Yam points out. "A useful example is a planet orbiting around a star." In this example, the principle of superposition of forces amounts to 'taking the system apart'. The mathematical backbone provided by calculus amounts to 'smoothness'. The

central roles of mass, gravitational constant and distance amount to the 'few important parameters'.

But this is not the case with complex systems. Rather, a complex system is "composed of simple parts where the collective behavior is complex," given a level of coarse graining and a context. Consequently, this "emergent complexity," as Bar Yam calls it, will not exist if the system is taken apart. It will not exist if only a few parameters are considered to be important. It will not exist if smoothness is assumed across the entire system. Of course, the notion of coarse graining is related to smoothness, but coarse graining is not equivalent to it. Rather, smoothness is a specified level of coarse graining. For example, an assumption of continuity in an orbital state vector is a coarse graining that is isomorphic with the real-number line. But in the study of complex systems, it is not useful to coarse grain down to a level by which local information is 'smoothed over'. So simple systems can be a starting point for studying complex systems, but the paths will diverge as these technical details are probed.

Technical Approaches to Studying Complex Systems

Where do the tools used to describe simple systems diverge from those useful for studying complex systems? Bar Yam explains, "[the] origin of simplicity is an averaging over the fast microscopic dynamics on the time scale of macroscopic observations ... and an averaging over microscopic spatial variations." This is the *ergodic theorem* and it is at the root of the 'smoothness' assumptions used to study simple systems. "Equilibrium systems are divisible and satisfy the ergodic theorem," such as the planetary orbit example mentioned earlier. But "[complex] systems are composed out of interdependent parts and violate the ergodic theorem. They have many degrees of freedom whose time dependence is very slow on a microscopic scale." This lack of an equilibrium end-state for complex systems is one of the thorniest issues governing their study. This is why the formation of a schema plays such a central role in complex systems research. There is no

‘end-state’ description obtainable by propagating time-dependent equations forward in time to infinity (e.g., differential equations) [3][4][5]. Rather, the study of complex systems is punctuated by formation and re-formation of a schema. A schema must be repeatedly updated when given new evidence to describe the “interdependent parts” about which it receives information.

Here we see a major break from the way simple systems are studied; simple systems – although they may contain many parts – can be described by one schema that is valid as time moves forward and more information is produced by the system. Returning to an earlier example, the laws of motion form the schema used to describe planetary orbits around a star. This schema remains valid as time moves forward as a consequence of the ergodic theorem; however, complex systems cannot be studied this way. A complex system is studied by the formation of a schema whose accuracy is repeatedly checked against the available information – or, the evidence.

Nonetheless, it must be recognized that there is a problem in complex systems research. On the one hand, complex systems research employs a schema to condense information but, on the other hand, this schema cannot be obtained by traditional methods of condensing information, such as smoothness and superposition. Instead, the spatial and temporal richness must be carried forward in time as the description of the complex system evolves. But in seeking the *crude complexity*, i.e., the shortest possible description of a system, as Gell-Mann calls it, a primary goal of complex systems research is simultaneously to condense this richness. With this conundrum in mind, it is possible to look for ways to walk this fine line between both the richness and the compression of a description.

Our primary clue in schema formation, as Gell-Mann points out, is information and so the information provided by a complex system is where the means to its understanding begins. The information plays a central role in what Bar Yam calls the focus of all efforts in complex systems: “to map a system onto a description of the

system." Bar Yam continues, "to understand the relationship of information to systems, we must also understand what we can infer from information that is provided. The theory of logic is concerned with inference. It is directly linked to computation theory, which is concerned with the possible (deterministic) operations that can be performed on a string of characters [.]" Bar Yam thus points us to computation theory to study information about a complex system. Specifically, this field of knowledge leads to "a set of models that capture aspects of the dynamics of simple or complex systems." Necessarily, we shall focus on how these models capture the dynamics of complex systems, rather than simple ones.

Considering our caveat on smoothness though, it is necessary to recognize that there will be a sampling rate of the information that is modeled. This will necessarily affect the choice of models used to capture the dynamics of the system in question. As Bar Yam points out, "treatment of dynamics will often consider discrete rather than continuous time." Not only is smoothness too strong of an assumption for complex system descriptions but also "computer simulations are often best formulated in discrete space-time variables with well-defined intervals." Furthermore, this type of sampling lends itself well to the formation of a schema because this does not happen continuously, in terms of infinitesimal time slices. Rather, a schema is formed over a discrete time interval and then updated upon subsequent ones. During each time interval, more information is observed that this leads to a new schema.

Enter Biology

In the task to form a schema much has come from biology. Gell-Mann already has pointed this out and Bar Yam considers the same idea as well. In fact, the reason for summoning biology comes from the potential difficulty of describing a complex system. In *Complex System Dynamics* [6], Weisbuch points out the potential pitfalls of using computer simulations to describe a complex system: "for many computationally intensive

... tasks, the difficult of programming can be avoided by a learning technique. Instead of listing the series of operations to be executed by the machine, it suffices to present it with some examples, along with some general guidelines about how they are to be treated." From this focus on learning, Bar Yam continues the thought to consider mathematical models based on the function of the brain:

> The functioning of the brain as part of the nervous system is generally believed to account for the complexity of human (or animal) interaction with its environment. The brain is considered responsible for sensory processing, motor control, language, common sense, logic, creativity, planning, self-awareness and most other aspects of what might be called higher information processing. The elements believed responsible for brain function are the nerve cells – neurons – and the interactions between them.

Recalling the "fight-fire-with-fire" approach of complex systems, it seems like a good idea to use the brain as a basis for understanding another complex system. However, there is a problem: "A variety of mathematical models have been described that attempt to capture particular features of the neurons and their interactions. All such models are incomplete." This difficulty in modeling a brain from its fundamental computational units has been a primary inhibiting factor. Nevertheless, there are clues that come from looking at how neurons interact, specifically focusing on interactions during *learning*. Generally speaking, learning is the process of gathering and comprehending information. Consequently, it bears a strong resemblance to schema formation, so it seems reasonable to study it biologically here.

There has been considerable research on learning over the years, but there is one idea that consistently resurges: Hebbian learning. Hebbian learning – or Hebbian imprinting – is considered to be the "principle mechanism for adaptive learning." This

notion of learning "suggests that when two neurons are both firing at a particular time, an excitatory synapse between them is strengthened ... Thus, the imprinted pattern of neural activity becomes a memory." Extending this line of thought across all of the brain's neurons, this means that information to the brain is condensed into these imprints – or memories – ready to be summoned by excitation of one or more neurons in this chain of neural activity. The expression 'cells that wire together, fire together' summarizes this proposed mechanism. Though it is an over-simplified account of learning, it illustrates some key principles that could be of use in the study of complex systems. Specifically, recalling the need in complex systems to condense observed information into a schema, it seems that learning mechanisms found in the brain could be of tremendous use in doing so.

With this motivation, much research has been done to implement learning via a machine. Some implementations have sought to directly model neural dynamics. For instance, Bar Yam points out one attempt with a type of neural network called an *attractor network* that "maps a variety of patterns onto an imprinted pattern." He goes on to say, [this] is equivalent to a classification of patterns by a category label ... Classification is also a form of pattern recognition." So classification via models of neural dynamics might provide an avenue for condensing information witnessed in time into a schema.

It is also important to consider regions of the brain, rather than just individual neurons, when studying learning. As Bar Yam puts it, "[to] neglect the description of the subdivisions of the brain and try to explain brain function directly from the behavior of individual neurons would be to skip an important and simplifying level of description. One of our primary tasks, therefore, in studying neural networks, is to investigate and identify the function and interaction of subnetworks." Though Hebbian imprinting suggests a path for exploring information condensation, it is the regional effect of this mechanism that is also important for classification and pattern recognition. Specifically,

different sub-levels of neural networks perform functions that are comparable to those observed in the brain during learning. For instance, Bar Yam discusses a multilayer feed-forward network, in which "the input later (or first few layers) represents sensory processing, and the output layer (or last few layers) represents motor control." This mapping between regions of the brain and regions – or sub-levels – of neural networks becomes a useful guide when comparing the abilities of these two complex systems in schema formation. As Bar Yam explains, the information flow when training neural networks generally maps to that of brains when learning.

An illustrative example of this mapping comes from analysis of how the visual system would map onto a neural network. Bar Yam considers a scenario in which information about color, shape and motion of objects in a receptive field are available. This information is to be used by the neural network to identify distinct objects. "Within each of these attribute categories we can construct a list of attributes ... The existence of three attribute categories enables a large number of descriptive categories to be constructed. A description is composed out of a selection of one attribute from each category." So if one neural sub-network categorized color information, and another categorized shape information, and still another categorized motion information, then the selection of attributes from each of these categories leads to an identification of an object based on a constructed list of attributes. Furthermore, the description – or the schema – of the object emerges from this list. Here we see the connection between categorization and description.

Biological processing mechanisms comprehend information via categorization across many channels. These categorizations then allow for a description of the information being observed. Consequently, it is has been of fundamental interest in the study of complex systems to employ these abilities computationally. But as Bar Yam and many others agree, "[intermediate] layers are less clearly identified" when comparing neural networks to biological networks of neurons. It is therefore of prime importance

when employing biology as a motivator for computation to pay attention to this mapping. Specifically, we must know where the analogies hold and where they are insufficient because this knowledge will determine the efficacy of the schema that is constructed.

In summary, the study of a complex system relies on its description. For systems that are complex, these descriptions tend to be quite long. However, taking cues from biology appears to open a path of inquiry in the study of complex systems along which learning plays a central role. Specifically, categorization and pattern recognition play fundamental roles in providing a means to describe something. But the specification of how a method of schema formation maps to its biological inspiration is also important, since this determines the accuracy of the pattern recognition. Consequently, these avenues of inquiry are worth considering in the study of complex systems and they motivate the need to delve more into machine learning.

Situational Awareness

Before machine learning is probed though, let us look at the study of complex systems from the human side. According to the literature surveyed thus far, complex systems research generally focuses on directly quantifiable ways to accomplish their study. Neural networks are an example of this. But there is much insight to be offered by human factors research, especially as it pertains to *situational awareness*. Just as brains have motivated research into machine learning, observations on the situational awareness of humans in dynamic and uncertain environments provides another avenue of inquiry in the study of complex systems.

Situational awareness (SA) research has been done in many contexts. Substantial research began with the SA of pilots in cockpits. For instance, pilots need to monitor and to integrate much information in order to control an airplane. Other research has looked at operators of complex manufacturing systems, e.g., nuclear power plants and refineries

[7]. In each of these tasks, humans must monitor and integrate data into an evolving schema used to make operational decisions.

One particular context in which there has been much research is in the decision-making of tactical commanders. And as information technologies have grossly affected the pace and style of war in the 21st century, there has been more of a need to understand tactical decision-making on a deeper level. This new type of warfare is called *Network-Centric Warfare* (NCW). In *Network Centric Warfare: Developing and Leveraging Information Superiority* [8], its authors summarize the thoughts of many researchers in this field: "[Network centric warfare rests on] the ability to capitalize on opportunities revealed by developing an understanding of the battlespace that is superior to that developed by an adversary." This understanding necessarily comes from the information available about the battlespace. So the ability to execute network centric warfare rests firmly on a decision-maker's ability to understand this information. But classical decision theory methods do not suffice to understand this process. Consequently, experimental research in decision-making provides a useful insight and this ties back into situational awareness.

Building on the importance of information in such decision-making, there is also danger of information overload in the future. As Endsley points out, "[when the complexity of a system] exceeds human [mental workload] capabilities, situational awareness will suffer" [7]. In an increasingly electronic battlespace, with information growing at an exponential rate, it is inevitable then that human decision-making will need support. Otherwise, SA will suffer and tactical decisions will become flawed.

In network-centric warfare, two issues come together. The first is that its decision-makers must comprehend and act within an evolving and complex battlespace. The second is that this process will become exponentially more difficult as additional information is exponentially produced and becomes available. In the terminology of complex systems research, NCW decision-making depends on how a decision-maker

forms a schema. A good schema leads to good decision-making. In human factors, the formation and maintenance of a schema are crucial components for what is called situational awareness in that discipline. Consequently, we are necessarily concerned with a decision-maker's situational awareness in the context of network-centric warfare.

Just as biology has been a guide for machine learning, it can also be a guide for studying situational awareness. In order to see how humans do it, attention now turns to human factors research to adduce more guidance on how to be aware of a situation. The goal here is to find the elements that can lead to doing it computationally. Given the growing amount of information in NCW, and perhaps other applications, such an endeavor would ultimately support a decision-maker in such an environment.

Theories of Situational Awareness

Mica Endsley is a primary scholar in studies of situational awareness, and in her "Towards a theory of situational awareness" we find a useful starting point. She writes, "[in] addition to forming the basis for decision making as a major input, situational awareness may also impact the process of decision making itself. There is considerable evidence that a person's manner of characterizing a situation will determine the decision process chosen to solve a problem." From this we see that schema formation as part of situational awareness actually exists within a larger context. This context comes from a set of decisions needed to solve a problem. In other words, the formation and maintenance of situational awareness is a goal-oriented process. Endlsey goes on to say that situational awareness "involves far more than merely being aware of numerous pieces of data. It also requires a much more advanced level of situation understanding and a projection of future system states in light of the operator's pertinent goals. As such, situational awareness presents a level of focus that goes beyond traditional information-processing approaches in attempting to explain human behavior in operating complex systems." Here we see a similar process to the one that Gell-Mann raised (Figure 1),

except here the focus is on a human who is observing data and incorporating them into a schema. Moreover, Endsley reiterates Gell-Mann's idea of schema formation by mentioning the role of future state projection. The combination of developing a "situation understanding" and being able to project that into the future is what makes situational awareness such a unique information-processing approach. Furthermore, such an approach is necessary to give a person the ability to operate a complex system.

As mentioned earlier, there are many examples of situational awareness in complex systems operation. Despite the different applications though, "[acquiring] and maintaining SA becomes increasingly difficult, however, as the complexity and dynamics of the environment increase. In dynamic environments, many decisions are required across a fairly narrow space of time, and tasks are dependent on an ongoing, up-to-date analysis of the environment." Necessarily, as more information becomes available to the decision-maker, the constraints on making an informed decision become tighter. So it will be necessary – if it is not already – to aid the decision-maker with tools that can simulate what his/her situational awareness already does, but just on a faster and larger scale. This issue comes up in the literature on network-centric warfare too [8]: "Highly placed decision makers around the globe have noted the greatly increased pressures upon them to react quickly to breaking events ... Thus, the race is on. We need [to find] ways to respond more quickly with quality decisions." Specifically, a computational implementation of SA would make this possible – one that can handle increasing amounts of data with flexibility and process it at the speed of modern computers.

But first it is necessary to see how humans achieve SA in tasks that are more manageable for their cognitive abilities. This could perhaps then serve as a template for performing SA on a computer. Consequently, Endsley and other human factors researchers study decision-makers. For instance, starting with tactical commanders, Kaempf, Wolf, and Miller [9] found that "recognizing the situation provided the challenge to the decision maker." Endsley reads this result as further proof that SA is a

critical step to making a decision. For her, it is specifically the ability to *classify* a situation that makes a decision possible. Endsley reports, "[researchers] in many areas have found that expert decision makers will act first to classify and understand a situation, immediately proceeding to action selection" [10][11][12][12][13]. Another human factors researcher, Federico, corroborates the importance of categorization in decision-making in his analysis of expert versus novice naval officers [14]: "[professional] decision makers seldom use classical analyses and theories in naturalistic settings. ... [Once] they categorize a problem, they generate a highly likely alternative and evaluate its suitability to the current circumstance." So we see that there is substantial research that corroborates the importance of classification in decision-making.

There is further connection to what was discussed earlier about categorization and pattern recognition in the context of machine learning. Endsley reports [15][16][17][18], "Dreyfus (1981) presented a treatise that emphasized the role of situational understanding in real-world, expert decision making, building on [deGroot (1965), Mintzburg (1973), and Kuhn (1970)]. In each of these areas the experts studied used pattern-matching mechanisms to draw on long-term memory structures that allowed them to quickly understand a given situation." Here we see the vital importance of pattern recognition, or "pattern-matching" as Endsley calls it. Specifically, pattern formation during learning condenses the information that was learned into chains of neural activity that can be summoned when needed by a goal-directed decision. This is a fundamentally important concept to consider regarding how humans form a schema to condense observed information.

But no one person forms the same situational awareness as another. Endsley writes, "[Individuals] vary in their ability to acquire SA, given the same data input. This is hypothesized to be a function of an individual's information-processing mechanisms, influenced by innate abilities, experience and training. In addition, the individual may possess certain preconceptions and objectives that act to filter and interpret the

environment in forming SA." If every person forms a different SA then what useful common threads can be pulled from its study? With this question in mind, Endsley lays out a framework for SA that has become a standard point of reference in human factors research.

According to this framework, there are three hierarchical phases to forming and maintaining situational awareness. These phases are: Level 1 SA: Perception of the Elements in the Environment, Level 2 SA: Comprehension of the Current Situation, and Level 3 SA: Projection of Future Status. Figure 2 shows these three levels of SA (highlighted green), as well as how they fit into the larger context in which SA happens. While the whole chart is especially informative about the unique information-processing task that occurs in forming situational awareness, our focus is on the three levels of SA. In particular, we see here that Level 3 SA does not happen without Level 2 SA, which in turn does not happen without Level 1 SA.

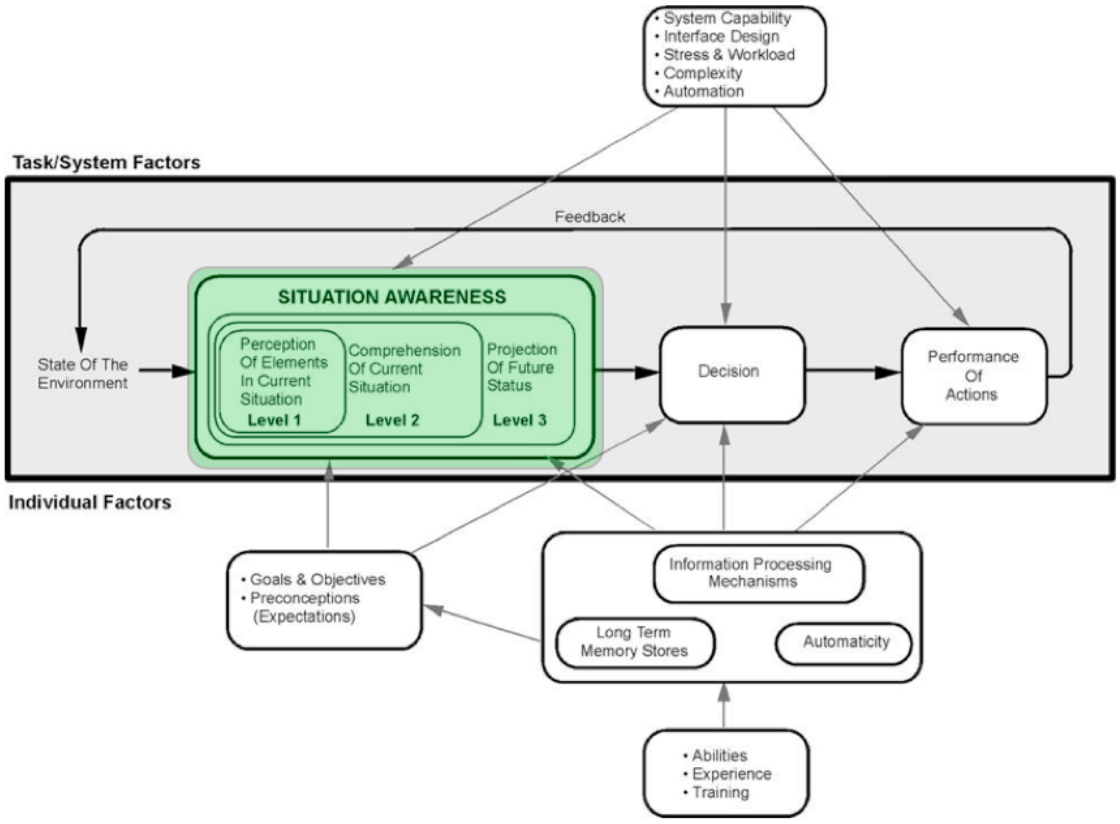


Figure 2: Model of Situational Awareness [7]

Endsley explains it in more detail:

> Level 2 SA goes beyond simply being aware of the elements that are present to include an understanding of the significance of those elements in light of pertinent operator goals. Based on knowledge of Level 1 elements, particularly when put together to form patterns with the other elements (gestalt), the decision maker forms a holistic picture of the environment, comprehending the significance of objects and events. ... The ability to project the future actions of the elements in the environment – at least in the very near term – forms the third and highest level of SA. This is achieved through knowledge of the status and dynamics of the elements and comprehension of the situation (both Level 1 and Level 2 SA).

This breakdown of SA is incredibly insightful. If a computational approach to situational awareness is going to be created then it might include elements of this framework. As will be shown later, the approach proposed and tested in this research attempts to emulate these levels of SA.

There are yet other aspects to SA that are important to consider in light of this framework. The first is that SA is a state of knowledge, rather than a process to obtain that knowledge. Endsley writes, "[It] is first necessary to distinguish the term *situation awareness*, as a state of knowledge, from the processes used to achieve that state. These processes, which may vary widely among individuals and contexts, will be referred to as *situation assessment* or as the process of achieving, acquiring, or maintaining SA." To compare this with complex systems terminology, SA is the schema regarding a given system's dynamics. And situation assessment is the process by which that schema is formed. Necessarily, with many ways of performing situation assessment there are equally many ways to form SA. So no given SA will be 'correct', as with the solution to a differential equation. Rather, SA is an evolving state of knowledge whose value is measured by its utility to a specified goal.

Another element to SA worth mentioning is its context dependence. Complex systems researchers noted this issue when defining the complexity of an observed system. But context dependence appears in Level 1 SA directly. Given the hierarchical nature of SA, context dependence necessarily affects higher levels of SA. Endsley writes, "[Elements] are specific to individual systems and contexts, and as such are the one part of SA that cannot be described in any valid way across arenas." This means that SA used for dynamic environment x is in most cases not valid for dynamic environment y. Since the elements that make up x are different from those that make up y, the Level 1 SA of each environment will be different. Consequently, the higher levels of SA will also be different. Furthermore, recalling issues of coarse graining, it could be that a different coarse graining of the same dynamic environment leads to a different SA than what one

had with the original coarse graining. Therefore, it is of utmost importance when defining the context to select data that leads to SA in light of some goal. This will also become important when implementing SA computationally.

One other aspect to SA that has only been implied thus far has been the role of time. As Endsley writes, "[Although] SA has been discussed as a person's knowledge of the environment at a given point in time, it is highly temporal in nature. That is, SA is not necessarily acquired instantaneously but is built up over time. ... [This] knowledge includes temporal aspects of that environment, relating to both the past and the future." This is a crucial point and we see a similarity to what was mentioned earlier about schema formation and selection. SA is built over time but when presented with a goal and evidence it must be summoned and used. Furthermore, the SA that is summoned at that moment has encoded into it the temporal aspects of the environment from the past, with expectations about future events. Necessarily, the process of building SA happens over time. "Thus it takes into account the dynamics of the situation that are acquirable only over time and that are used to project the state of the environment in the near future," Endsley continues. So SA must be acquired over time, but necessarily as more information is observed in time SA must condense this time-dependent data.

A final aspect to consider in the formation of SA is attention. As Endsley points out, attention links back to goals and the sampling rate of new information [19]: "In complex and dynamic environments, attention demands resulting from information overload, complex decision making, and multiple tasks quickly exceed a person's limited attention capacity. Operators of complex systems frequently employ a process of information sampling to circumvent this limit. They attend to information in rapid sequence following a pattern dictated by the portion of long-term memory concerning relative priorities and the frequency with which information changes (Wickens, 1992a)." What we see here is that attention acts as a filter to information deemed extraneous in light of an operator's goals. These goals guide attention to elements that are then

perceived in Level 1 SA at some frequency. In other words, there is an expectation built over time that information will change in the future at a comparable rate to that of the past. Although the mechanisms will be different, the importance of a sampling rate here is similar to what Bar Yam describes in the modeling of complex systems. He noted that well-defined time intervals are essential to computer simulations, so we can see another overlap between how SA is formed and the abilities of computers: in both cases, information is sampled at a rate and not continuously according to an assumed smoothness of time. But there is certainly inspiration to be drawn from how humans do it, given their observed success at doing so in many cases.

The final aspect to consider in SA is the role of goals. The goal-oriented aspect to SA has been mentioned all along, but some more detail about it would reveal its role in the levels of SA. When considering the role of goals in SA, it is unavoidable to identify the top-down and bottom-up processing that is occurring simultaneously. Endsley cites other human factors research in light of her hierarchy:

> In what Casson (1983) has termed a *top-down process*, a person's goals and plans direct which aspects of the environment are attended to in the development of SA. That information is then integrated and interpreted in light of these goals to form Level 2 SA. ... Simultaneously with this top-down process, *bottom-up processing* will occur. Patterns in the environment may be recognized that will indicate that new plans are necessary to meet active goals or that different goals should be activated.

What we see here is that attention as directed by goals also plays a huge role in how SA is formed. The goals determine what information is incorporated into a schema, but unsuspected patterns can affect the goals from the bottom-up. If SA is to be done

computationally then it must incorporate this combination of bottom-up and top-down processing that exists in humans.

In summary, we have seen that situational awareness is a unique information-processing mechanism employed by humans in the operation of complex systems. These systems can be mechanical, industrial or even human-based, but across this broad spectrum of examples the common thread is situational awareness. This SA is formed hierarchically with the combination of pattern recognition in time and goal-oriented data gathering in a specified context. Furthermore, these ideas overlap substantially with the surveyed complex systems research. This indicates that both complex systems and situational awareness research are studying a related problem. Necessarily, insights from both will strengthen that problem's understanding and possibly lead to its solution. That problem then is how to understand the evolution of a complex system, whether by schema formation or SA or more explicit forms of modeling. With that general problem identified, it would be helpful to specify the context now to a pertinent problem in engineering. Our attention now shifts therefore to the context of network-centric warfare.

Network-Centric Warfare and Decision-Making

One of the primary challenges in analyzing network-centric warfare (NCW) is matching the terminology of the military to that of scholarly research. Just as with complex systems and human factors though, the military community has a similar problem in the context of network-centric warfare to what we have surveyed thus far in more general contexts. In short, the problem is that there is a lot of information available and decision-makers need to make sense of it. As with complex systems, there is no better way to engage the material than to proceed directly into its heart and so this is where we begin with NCW.

Just as with complex systems research, network-centric warfare is an information-based process. In *Network Centric Warfare: Developing and Leveraging Information*

Superiority, its authors write, "NCW recognizes the centrality of information and its potential as a source of power. This potential is realized as a direct result of the new relationships among individuals, organizations, and processes that are developed. ... It is the cumulative impact of new relationships among warfighting organizations that are the source of increased combat power." In complex systems, information was vital to making a schema. For SA, it was vital input as well. In network-centric warfare (NCW), information is a source of power. And so we see how the goal-oriented nature of NCW establishes itself from the beginning. In particular, military supremacy is tied to information comprehension in NCW. In *Information and Knowledge Centric Warfare: The Next Steps in the Evolution of Warfare* [20], researchers from the U.S Air Force Research Laboratory point to an *Aviation Week* [21] article in which it is said that the determinants of success on the 2030 battlefield "will not be aircraft, ships or tanks, but rather the exploitation of knowledge and speed of execution based on that knowledge." So it is seems without question that information will have a central role – if it has not already – in military conflicts of the future. This is necessarily a concern to a decision-maker in such a context.

The key issue in network-centric warfare then is how available information becomes useful knowledge. Some clues come from the NCW literature. For instance, Alberts and colleagues write, "NCW is about human organizational behavior. ... NCW focuses on the combat power that can be generated from the effective linking or networking of the warfighting enterprise." So there is something of importance in how elements of NCW are linked together. This linking is what can then lead to combat power. They go on to say that commanders are "[empowered] by knowledge, derived from a shared awareness of the battlespace and a shared understanding of [other] commanders' intent ... [especially] when operating autonomously." Here we see that goals, or intent, are important as well as a shared awareness. Obviously, there is some overlap here with SA, especially via the role of knowledge in light of common goals. But

it is not only the awareness and goals of commanders that are important. Rather, "[there] needs to be equal emphasis placed upon developing a current awareness of both friendly and enemy dispositions and capabilities, and in many cases, there needs to be increased emphasis on neutrals." Once again, it seems that research in SA can have significant impact on these needs of the military. Indeed, Alberts et al. elucidate this clearly: "Battlespace awareness results from the fusion of key elements of information which describe or characterize the battlespace. ... The difficult comes in placing the information in a larger context and understanding its implications." Clearly, SA in network-centric warfare is termed *battlespace awareness*. And we get a glimpse here of how it happens. Specifically, there is a fusion of information that occurs. This is almost exactly what Endsley and others describe for building/maintaining SA. This is also very much like schema formation in complex systems research. What we see here therefore is that SA amidst network-centric warfare is really a type of complex systems problem.

Just as with complex systems research though, there is possibility for information overload when building NCW SA. Additionally, recalling from earlier, there is less time for decision-makers to process and react to unfolding events. As Alberts and colleagues point out, "[the] potential for information overload is real and great care must be taken to make sure that what is provided [to a decision-maker] is actually information and not noise. In addition, access to tools and expertise will be required to achieve battlespace knowledge." This provides additional evidence for how NCW SA is very much like the analysis of a complex system. Anyone can observe the unfolding events of a complex system, such as NCW, but the real need is to integrate that data into actionable information. In corroboration with what we have seen in SA literature thus far, Alberts et al. go on to write, "[what] is of value and what is likely to distract depend to a great extent upon what [an] entity is supposed to do." So we see here that the goal-oriented nature of knowledge formation cannot be ignored, just as it could not be ignored in the

formation of SA. But it is not the only determining factor. Rather, it is an influencing factor, just as it was in the formation/maintenance of SA.

Even though the analogies to complex systems research certainly exist for NCW, scholars in the field recognize that this is a perspective worth having. Alberts et al. summarize it as follows [8][22]:

> [It] is clear that our missions have gotten to be far more complex, and our challenges and adversaries less predictable. The information that we need to sort things out has gotten, simultaneously, more diverse and more specific. Our measures of merit have also become more varied and complex ... Dealing with this complexity will be a major challenge that requires approaching problems and tasks somewhat differently.

Even though the word "complex" appears in this quote, the connection to complex systems research is not superficial by any means. Rather, the goals – or missions – have become less predictable. Also, the information now comes in many different forms and its condensation into actionable knowledge remains a difficult task. Consequently, in light of surveyed literature on SA, shifting goals make the transfer of information into knowledge a prominent issue in NCW decision-making.

Survey of Available Analytic Tools

Having identified what is at stake in NCW decision-making, much research has been devoted to executing it by various analytic means. The AFRL researchers write, "[technologies] that will support knowledge-centric warfare are those that deal primarily within the cognitive domain and upper levels of sensemaking." So the focus is on which technologies can relate to cognitive information processing. Specifically, these researchers identify many areas, all of which have inter-related needs and goals. Some of

these areas are as follows: predictive battlespace awareness, knowledge reasoning, multi-domain information fusion and knowledge discovery. For instance, for predictive battlespace awareness, they go on to list some required technologies:

> a) [Real-time] assessment of adversarial intent ... b) decision theory (... By relaxing the classical assumptions of perfect rationality and perfect foresight, we obtain much improved explanations of initial decisions, dynamic patterns of learning and adjustment ... such technology would greatly improve the anticipation of adversary actions); and, c) real-time Bayesian inferencing which provides a different approach to the estimation of adversarial response [.]

Even though the literature on SA has shed some light on useful decision theories and assessment of intent, Bayesian inference is a somewhat new idea. It will be revisited in more detail in subsequent sections. For knowledge reasoning, the authors see a "need to develop automated capability to reason, infer and discover knowledge implicit in extracted information [through knowledge discovery and machine learning tools]." So we see here some overlap with what we had encountered in complex systems research regarding the utility of machine learning again. For multi-domain information fusion, "[new fusion] approaches are needed to reduce information ambiguity from multiple sensor types and geo-locations." This is the first mention of the discipline of data fusion thus far. But its presence has been implicit in discussions on schema formation and Level 2 SA, especially if these processes are to be done computationally. Finally, knowledge discovery requires similar technologies to knowledge reasoning. In particular, "[knowledge discovery] technology deals with machine learning, case-based reasoning, similarity metrics and pattern learning. Data [mining and text mining] are subsets of [knowledge discovery] ... There exists a need to develop the necessary machine learning technologies to enable a system to learn from example instances consisting of data

entities, relationships and their attributes, and models of scenarios of interest." With all of these avenues of technological development in NCW decision-making, we see some common threads. In particular, there is a need to extract meaning from data and some of the chief ways of doing this are machine learning, data fusion, Bayesian inference and pattern learning. So we see even more how NCW is very much a complex system in this regard as well. Specifically, complex systems are analyzed effectively with these tools and NCW is no different.

Much of this discussion on NCW and how it relates to various techniques of operating within it can be summarized visually. Figure 3 presents a network-centric military structure in which information – or "infostructure" – is the sine qua non.

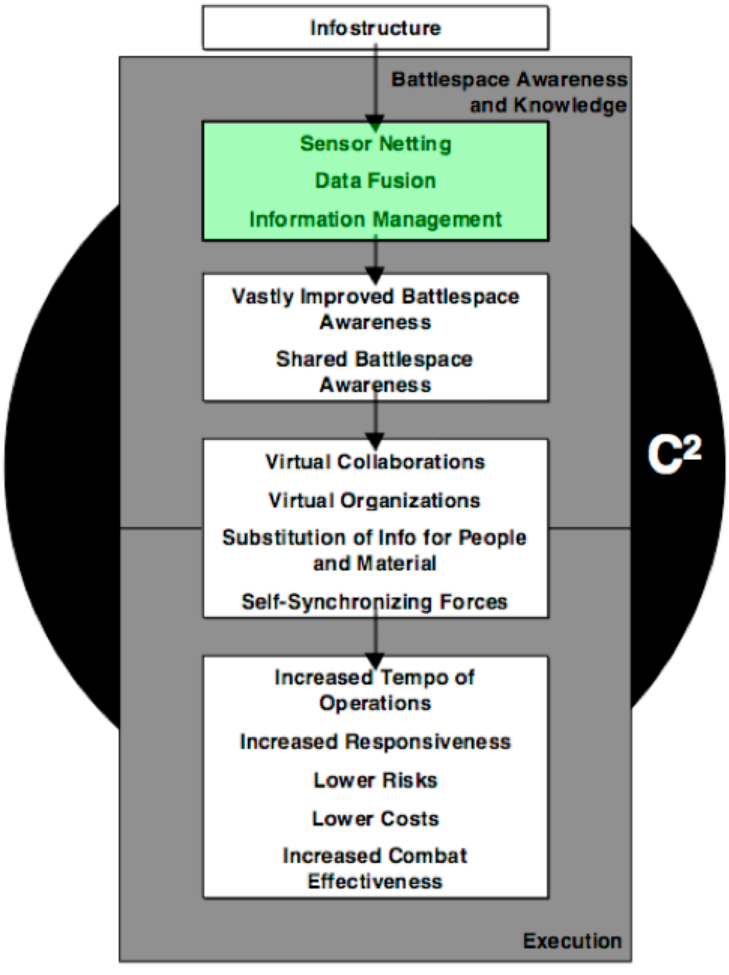


Figure 3: The Network-Centric Military [8]

This information's fusion into actionable awareness then feeds down into how the military executes objectives within command & control (C^2). Just as situational awareness existed within a larger context, the "sensor netting [and] data fusion" step in network-centric military decision-making also exists within a larger structure. In this work, we are specifically focusing on this element to NCW. In particular, we are examining how to get from the information – or "infostructure" – to a good SA about NCW – or the "vastly improved battlespace awareness." This is what complex system researchers examine in their studies, and this is what happens in the SA of someone operating a complex system. The only difference is that NCW is now the specified system or context. Having surveyed this specific complex system and its relevant literature, we now return to machine learning and other techniques by which it can be studied as a complex system.

Machine Learning, Data Fusion and Data Mining

Machine learning, data fusion and data mining are very similar endeavors in computing. Specifically, the goal of each is to extract useful information from data. While one can be an implementation of the other – such as machine learning's relationship to data fusion or mining, the discussion of one usually cannot help but overlap with any other. So we will just begin with some broad features extracted from the relevant literature and sequentially target certain techniques that can be of use in light of our surveys of situational awareness, complex systems and network-centric warfare.

Data Fusion

Given the importance of data integration in the formation of situational awareness, data fusion would be an appropriate starting point to examine its computational execution. Much work has been done on data fusion as it has developed

into a discipline of its own. The first widely known and comprehensive survey of data fusion was "An introduction to multisensor data fusion" by Hall and Llinas [23]. Much of the work in this field can be gleaned from this survey and so this will be our starting point.

First of all, there is a breadth of applications for data fusion. Hall and Llinas point out, "[Multisensor] data fusion is an emerging technology applied to Department of Defense (DoD) areas such as automated target recognition, battlefield surveillance, ..., and to non-DoD application such as monitoring of complex machinery[.]" Similar there is a large amount of work from different areas that has contributed to its development. In particular, "Techniques for multisensory data fusion are drawn from a wide range of areas including artificial intelligence, pattern recognition, statistical estimation, and other areas." Even though this survey of data fusion is from 1997, there have bee developments in it since then. Although it will not be possible – or necessary – for this research to survey all of these efforts, some examples are worth mentioning in the context of network-centric warfare decision-making and so these will be mentioned in due time. Before then, it is necessary to see why and how data fusion work will be of use in light of what has been found from surveys of SA and complex systems research thus far.

Data fusion puts a computational spin on the crucial steps of situational awareness formation. In particular, Hall and Llinas note the following [24][25][26]:

> Data fusion techniques combine data from multiple sensors, and related information from associated databases, to achieve improved accuracies and more specific inference than could be achieved by the use of a single sensor alone [.] The concept of multisensor data fusion is hardly new. Humans and animals have evolved the capability to use multiple senses to improve their ability to survive. Thus multisensory data fusion is naturally performed by animals and humans to

> achieve more accurate assessment of the surrounding environment and identification of threats, thereby improving their chances of survival.

Although no exact computational mechanism is described here, it is clear from this comparison to biological systems that there will be substantial overlap with what has been seen thus far from human factors research on situational awareness. This is a point that was not mentioned explicitly in the surveyed SA literature. Specifically, SA is formed and maintained not just by perception in one sensory channel. In fact, with more information available from more sensors, it follows that better SA can result from multisensory data fusion if the information-processing mechanism is flexible enough to handle the increased data flow. Hall and Llinas corroborate this line of though as well: "[There is] statistical advantage gained by combining same-source data (e.g., [by] obtaining an improved estimate of a physical phenomenon via redundant observations)." We can glean from this discussion then that in data fusion, there are channels by which information comes in to a computer. There is not just one channel of information. Rather, it enters somewhat compartmentalized by the apparatus used for observation.

Most importantly, the ability to attempt data fusion computationally is substantially within reach, given modern technological abilities. Hall and Llinas cite [27][28], "While the concept of data fusion is not new, the emergence of new sensors, advanced processing techniques, and improved processing hardware make real-time fusion of data increasingly possible ... [Recent] advances in computing and sensing have provided the ability to emulate, in hardware and software, the natural data fusion capabilities of humans and animals." So then which techniques can be employed to engineer data fusion? Hall and Llinas continue [29][30][31], "Techniques to combine or fuse data are drawn from a diverse set of more traditional disciplines including: digital signal processing, statistical estimation, control theory, artificial intelligence, and classic

numerical methods [.]" Expectedly, we see already some overlap with methods employed in complex systems research for quantitative analysis.

But since data fusion is specifically focused on the computational implementation of information analysis from multiple domains there are exact questions to be answered in the course of its accomplishment and steps to be taken when it is done. Hall and Llinas lay out a series of questions and a process (the JDL Data Fusion Process) that have become widely referenced in data fusion literature. "The fundamental issues to be addressed in building a data fusion system for a particular application include:

1. What algorithms or techniques are appropriate and optimal for a particular application;
2. What architectures should be used (i.e., where in the processing flow should data be fused);
3. How should the individual sensor data be processed to extract the maximum amount of information;
4. What accuracy can realistically be achieved by a data fusion process;
5. How can the fusion process be optimized in a dynamic sense;
6. How does the data collection environment ... affect the processing;
7. Under what conditions does multisensor data fusion improve system operation?"

As routinely happens when one wishes to translate seemingly simple mechanisms into a computational implementation, some issues not explicitly formulated before rise to the forefront. For instance, when discussing the formation and maintenance of SA, there was no mention of where in the processing flow data should be integrated into a schema. It was just implied that this somehow happens. But when data is recorded computationally, this issue becomes important. This of course is a related issue to the third question above

regarding data processing. This also did not arise explicitly in the SA literature because humans have developed biological data processing mechanisms to translate visible light, smells, tastes, sounds and tactile pressures into a format suitable from the brain's information processing abilities. But in doing so with computers, this issue can be dealt with in a myriad of ways. As we see from just these two questions out of the seven listed above, data fusion techniques shed light on the details of how one would implement SA computationally.

In reaction to the needs of data fusion as expressed by these questions, the Joint Defense Laboratories (JDL) Data Fusion Process was developed. As Hall and Llinas write, "The JDL Data Fusion Process model is a conceptual model which identifies processes, functions [and] categories of techniques[.] ... [The] data fusion process is conceptualized through seven steps." These are not meant to answer the above seven questions on a one-to-one basis but as a whole they address the questions in their totality. The seven steps and a brief description of each are as follows:

1. Sources of Information – This step considers what channels of information are available. These channels could be input from "local sensors, ... distributed sensors linked electronically to a fusion system, ... [or] other data."
2. Human Computer Interaction (HCI) – This step allows human input to the data fusion process, such as "human assessments of inferences" generated by the fusion process.
3. Source Preprocessing – This step "reduces the data fusion system load by allocating data to appropriate processes[.] Source preprocessing also forces the data fusion process to concentrate on the data most pertinent to the current situation.."
4. Level 1 Processing: Object Refinement – This step "combines locational, parametric and identity information to achieve refined representation" of entities

of interest. An example entity of interest could be an object in the case of target tracking, or system state in the case of system analysis.

5. Level 2 Processing: Situation Refinement – This step "develops a description of current relationships among objects and events in the context of their environment." This is done to "determine the meaning of a collection of entities."
6. Level 3 Processing: Threat Refinement – This step is concerned with projection into the future. Specifically, this step "projects the current situation into the future to draw inferences about enemy threats, friendly and enemy vulnerabilities, and opportunities for operations."
7. Level 4 Processing: Process Refinement – This final step is what is called a *meta-process* because it lies somewhat outside and somewhat inside the data fusion process. This is so because this step is concerned with monitoring performance, identifying ways for improvement, determining source specific requirements and allocating sources to achieve mission goals.

Even though these steps are enumerated sequentially, there is no need for one to follow the other this way. Rather, each step works with the other in an integrated framework as illustrated by Hall and Llinas (Figure 4).

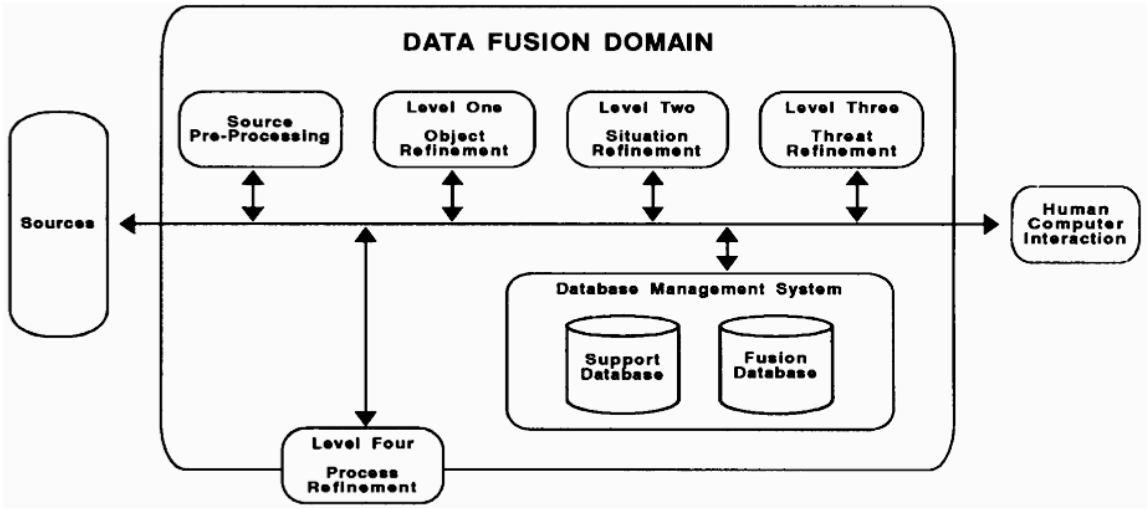


Figure 4: Data Fusion Process Model [23]

The data fusion process is quite similar to the highlighted part of NCW decision-making (Figure 3). It is also quite similar to Endsley's model of SA formation (Figure 2). Furthermore, it is related to Gell-Mann's description of how complex system analysis works (Figure 1). In all of these models, a series of steps are taken to extract meaning from data. Figure 4, however, gives a step-by-step account of what steps are needed to extract meaning from data pooled across a range of different sensory modalities. Consequently, this is of crucial importance in the translation of data into actionable knowledge.

Hall and Llinas go on to survey some technical methods for doing the above steps. Additionally, they propose a class of architectures that could be of use for various data fusion problems. However, none of these architectures fit close enough to the problem of NCW decision-making as it has been studied thus far. In fact, in "Information fusion for situational awareness" [32], researchers from the AFRL also agree, "While the JDL provides a functional model for the data fusion process, it does not model it from a human perspective." Consequently, these researchers choose an approach based on Endsley's framework, but do not culminate in a tractable implementation. A balance must therefore be sought in making a computational implementation of SA between the

general guidelines provided by Endsley and the step-by-step functionality proposed by the JDL.

Before finding this middle ground though, it is useful to finish the survey of data fusion by focusing on some explicit techniques used to do it.

Table 1: JDL Process and State-of-the-Art Techniques

JDL Process	Processing Function	Techniques
Level 1: Object Refinement	Data alignment	• Coordinate transforms • Units adjustments
	Data/object correlation	• Gating techniques [52] • Multiple hypothesis association probabilistic data association [63],[64] • Nearest neighbor
	Position/kinematic and attribute estimation	• Sequential estimation [19], [73], [75] - Kalman filter - αβ filter - Multiple hypothesis [79] • Batch estimation [69], [70], [71] • Maximum likelihood [80] • Hybrid methods [76], [77], [78]
	Object identity estimation	• Physical models • Feature-based techniques - Neural networks - Cluster algorithms [56], [57] - Pattern recognition [87], [88], [89] • Syntactic models
Level 2: Situation Refinement	Object aggregation Event/activity interpretation Contextual interpretation	• Knowledge-based systems (KBS) - Rule-based expert systems - Fuzzy logic [60] - Frame-based (KBS) • Logical templating [114], [115] • Neural networks [96], [97], [98], [99] - Blackboard systems
Level 3: Threat Refinement	Aggregate force estimation Intent prediction Multi-perspective assessment	• Neural networks - Blackboard systems [122] • Fast-time engagement models
Level 4: Process Refinement	Performance evaluation	• Measure of evaluation [2] • Measures of performance [2] • Utility theory [59]
	Process control	• Multi-objective optimization [59] - Linear programming - Goal programming
	Source requirement determination	• Sensor models
	Mission management	• Knowledge-based systems

Hall and Llinas have created a table matching each of the JDL Process steps to state-of-the-art techniques. This table has been reproduced here for convenience as Table 1 (note that reference numbers in table are those of Hall and Llinas). As mentioned earlier, these were state-of-the-art in 1997, and there has been much research in data fusion since then. However, Table 1 provides a good overview for what kind of techniques are of use in

data fusion. Other researchers have recognized similar techniques as important for data fusion as well [32]. Furthermore, the techniques mentioned in this table and other sources have evolved or have informed techniques that are state-of-the-art today.

To go more in-depth into state-of-the-art techniques today it is necessary to constrain the survey to those methods applicable to our context of focus, i.e. network-centric warfare decision-making. To do so, our attention will now turn towards some specific attempts at doing this. As we shall see, these attempts have used data mining and machine learning algorithms as ways to fuse the data.

Data Mining and Machine Learning in SA and Decision-Making

From the literature, there are at least three notable examples of situational awareness implementations to consider in the context of network-centric warfare or strategic decision-making. The first one is a framework for SA in the context of the first Gulf War. The second one is a coordinated machine learning decision support for perceiving threats to a base in a potentially hostile environment. The third one is an evolved neural network used in strategic decision support. As we shall see from each of these examples, data mining and machine learning feature prominently. Conversely, data fusion either is incorporated or omitted from these examples depending on the implementation. By studying these three examples, we will reveal how data mining and machine learning can assist in decision-making, especially within the NCW environment. Furthermore, we will see when data fusion can be of use and when it can be a hindrance to such a task.

Case 1: A NCW Framework for Situation Awareness

Although the authors of "Building a framework for situation awareness" [33] do not explicitly call this situation an instance of network-centric warfare, it most certainly is such a case. In this study, AFRL researchers consider an SA framework in the context of

the Iraqi incursion into Kuwait, which was a case in which signs were given about the potential course of events. Recalling from earlier, "[Network centric warfare rests on] the ability to capitalize on opportunities revealed by developing an understanding of the battlespace that is superior to that developed by an adversary." In this specific case of the first Gulf War, "One hundred and forty key events were identified from February 24, 1990, when Saddam Hussein threatened the Premier of Kuwait, through January 17, 1991, when the US began bombing Baghdad. The concern raised was Iraq's aggression towards its neighboring countries [.]" So the problem here was to interpret and fuse various elements of data that were observed between these time periods. From this process, the aim was to identify Iraq's intent towards its neighboring countries.

Given this scenario and its substantial scope, it shares many commonalities with what has been encountered thus far in complex systems research. As in that discipline, the need here is to form a schema that condenses the data on 140 witnessed events. Necessarily, since this scenario developed over the course of 11 months, a set of schemata would be formed as data was observed, and then new data would exert selective pressures on certain schemata according to Figure 1.

As the title of the just mentioned paper suggests, this scenario is a classic example of the use of situational awareness. It is interesting to note that the authors explicitly choose Endsley's SA model (Figure 2) as a starting point against the JDL data fusion process (Figure 4): "While the JDL provides a functional model for the data fusion process, it does not model it from a human perspective. Endsley provides an alternative to the JDL model that addressed Situation Awareness from this viewpoint." Using Endsley's model as a starting point, the authors propose an SA framework that maps Endsley's steps to actual analytic tools (Figure 5).

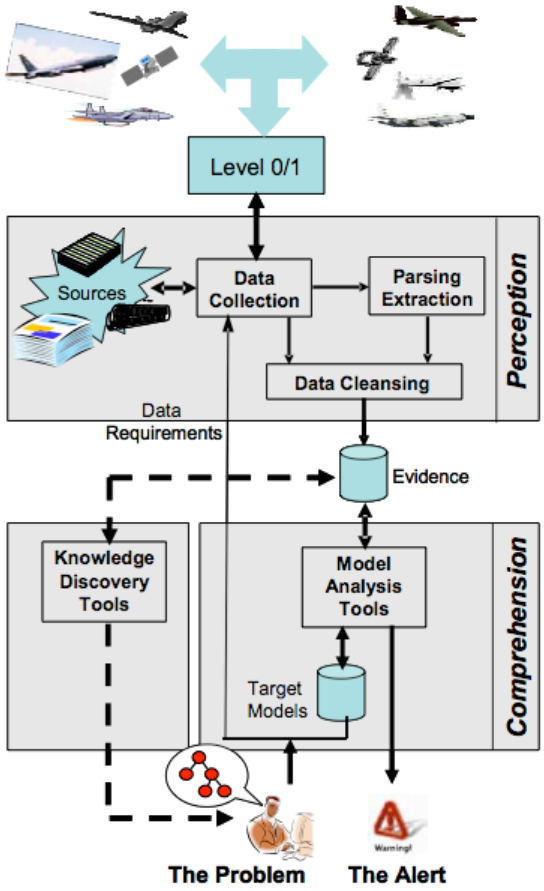


Figure 5: Situational Awareness Framework for NCW [33]

Of course, there are many similarities between this figure and the previous ones from SA, complex systems and NCW decision-making. The information is sensed and made ready in the "Perception" box to form evidence. This evidence is then fed into "Knowledge Discovery Tools." But what happens here? How does that then feed into "The Problem" to make "Comprehension" happen in that box? The authors explain as follows:

> In order to comprehend the current situation and its relevancy one must have some knowledge of similar situations that occurred in the past and relevant events currently occurring. If this prior knowledge does not exist, we need to learn or discover it. This knowledge can be captured as models [that] can be learned by deriving them through data sets and would include such concepts as ... group

memberships. This area is what we have called Knowledge Discovery Tools. One of the major areas that fall under this topic is Data Mining.

Knowledge discovery has been mentioned earlier in the survey of current analytic techniques available for NCW decision-making. Here we get another description of what it is as well. Furthermore, we can see now the connection between knowledge discovery and data mining directly. In other words, data mining in this context is done to extract relevant knowledge about situations similar to this incursion by Iraq into its neighboring countries. Of course, this is a very specific context and so only data pertaining to this situation would be useful as input to Knowledge Discovery Tools.

Knowledge discovery can of course happen without knowledge of similar situations in the past. This is because machine learning and data mining techniques can be utilized as data is revealed about the current situation. In *Machine Learning* [34], Tom Mitchell calls machine learning "the study of computer algorithms that improve automatically through experience." In the SA framework paper, the AFRL researchers cite Witten & Frank's text on data mining [35] to define data mining as "the extraction of implicit, previously unknown, and potentially useful information from data." Since data mining is usually implemented automatically via a computer, it is generally considered together with machine learning, although it need not be. Nevertheless, knowledge discovery happens with data mining, which in turn generally is done via machine learning.

Looking closer at data mining reveals some choices for how to implement knowledge discovery. "Data mining techniques can be divided into two activities: (1) identifying patterns based on event associations which we refer to as pattern learning and (2) identifying groups based on similar activities which we refer to as community generation." The first kind of data mining is reminiscent of what we encountered earlier with Hebbian learning. Here associated events could be linked into a pattern by

simultaneously firing neurons, for instance. The second kind of data mining is more concerned with categorization based on similarity. As already has been noted above, categorization and pattern learning are related computational activities. This is because the relations between various data form the patterns, but each pattern or group of patterns can be categorized.

When finding relations between data, the coarse graining matters. "It is crucial that we thoroughly sift through archived data to look for the associations between entities at multiple levels of resolution. Pattern learning technologies serve to address this task by providing techniques that mine *relational* data." But there are some challenges to learning relational data:

- "Relational learning must consider the neighborhood of a particular entity, and not just a singular record.
- Most learning is predicated on (usually false) assumptions of independent samples. Relational data does not meet this criterion.
- Data must be semi-structured to make learning possible."

As we develop the approach put forth later in this research, it will be seen how these challenges are met. But for now, they should be noted, especially in the context of complex systems analysis.

There is one more significant concern to note for relational data learning. This has to do with positive versus negative instances. The AFRL researchers cite artificial intelligence literature [36][37] in the context of their problem, generating an insightful train of thought:

> Jensen states that the biggest concern in developing a pattern learner for situation awareness is the relatively low number of so-called 'positive instances', turning

> the pattern learning process into an anomaly detection process. Problems such as these are often considered 'ill-posed' in the computational learning community, and more often than not, partially invalid assumptions about the data must be made to correct for these conditions. ... While the challenges are significant, so too is the potential payoff. Relational learning allows systems to exploit multiple tables in a database without the loss of information that occurs in a join or an aggregation. The resulting discoveries may include predictive patterns that more accurately describe the world by utilizing entities' attributes as well as the relationships between entities in the learning process.

Although this is quite a long train of thought to follow, it lays out many important points for employing data mining techniques that exploit relations in the data. First of all, the fact that such problems are usually ill-posed is centrally important. Second, to compensate for this, slightly invalid assumptions about the data are employed to build patterns of relation. While such a process will produce inaccurate predictions sometimes, there is a substantial chance of discovery because much information is preserved (i.e. not aggregated) in a computationally manageable quantity. Consequently, these points are a prime source of motivation for the approach to computational SA explored in this work.

Although not every aspect of Endsley's SA model is accounted for, the SA framework done in this study was top-down rather than bottom-up. The authors write, "The process begins by first defining the problem in terms of a model. The model is a simple acyclic graph specified in XML. ... We note here that these interrelationships are simple and purely hierarchical. At the lowest level of our model are the indicators or actual events/observations. These indicators bind the conceptual and computational worlds together." A sample model is shown in this work and it provides a useful visual aid (Figure 6). Recall that "indicators" serve as the information into the model, i.e. the 140 witnessed events. These indicators are then connected by relationships between

them. Since this is a top-down implementation, the relationships are specified *a priori* via the acyclic graph. This sample framework for implementing SA gives a useful starting point for considering how to implement SA on a computational level.

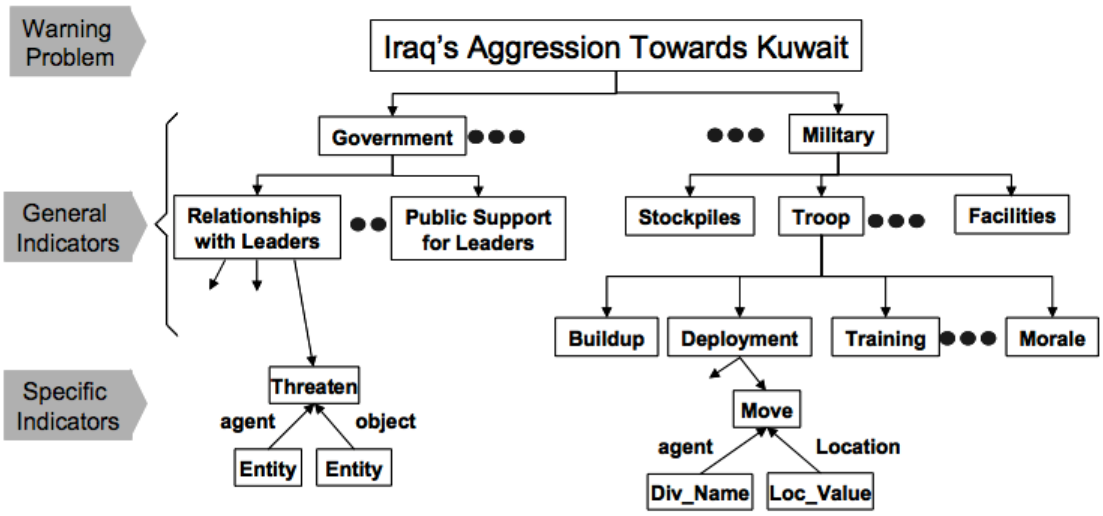


Figure 6: SA Framework Applied to Iraq Aggression Scenario [33]

It is particularly useful because it is also an example of how SA is crucial in a network-centric warfare scenario.

Before considering some shortcomings of this implementation, there are some worthy points noted in the authors' evaluation of SA frameworks in general. First, the authors write, "The success of any Situation Awareness system depends upon understandable Measures of Performance (MOP) and Measures of Effectiveness (MOE). These measures must include quantitative and qualitative characterizations and be directly tied to the mission of the system in question." This is an important point because SA is not a process that is easily quantifiable. By its very nature, SA is acquired through evolutions as a schema or set of schemata is refined. Furthermore, SA's value is judged in relation to goals. Though the authors give no specific indicators for what these MOPs

or MOEs would be, they note that these measures can be both quantitative and qualitative.

It remains to be seen exactly what suitable MOEs or MOPs would be; however, the authors give some indication about what they would be for their SA framework:

> At an abstract level the system may be viewed as a black box classifier. ... However, the difficulty arises in understanding these results. In order to accurately characterize the system, one must have a technique to characterize the input to the system. Such a technique must not only capture the differences between various test datasets, but also between test datasets and the real world.

As would be expected from previous discussion on the role of classification for decision-making, this seems to be a suitable route to focus on MOPs related to classification. Regarding input to the system, the authors describe a point that is similar to the issue of coarse graining. Specifically, how the datasets differ from each other and the real world is an important issue to consider when designing a computational SA. There is no indication given in this article regarding how effective the proposed top-down model was in this regard.

Necessarily, there are some shortcomings with this approach. Some of these the authors freely admit, such as the lack of bottom-up processing. But most notably, the events that serve as input to the framework have relationships imposed on them *a priori*. Despite what was said earlier about the utility of data mining techniques for relational data, this implementation of SA assumed the relations from the beginning. While this is certainly a consequence of the framework being top-down, it necessarily disallows any chance of knowledge discovery that can come from learning algorithms. With this shortcoming in mind, a more sophisticated implementation will now be discussed. Although it is cast as more of a decision support tool, rather than a computation

implementation of SA, many of the issues encountered in both tasks are similar, as we have seen. Our attention now shifts to this example.

Case 2: A NCW Decision Support Implementation

Brannon and colleagues [38] are the authors of "Coordinated machine learning and decision support for situation awareness." One of the most attractive features of this example is that it is analytically quite sophisticated. While the previous example proposed a framework and gave mild description of an XML-based acyclic graph, this implementation probes more into machine learning, data fusion and data mining principles. Necessarily, this will open the door into the uses of machine learning in NCW decision-making scenarios.

The authors begin with a brief survey of machine learning techniques and their relation to situational awareness problems. Specifically, the authors make a notable point on the lack of applications thus far of neural networks to SA problems: "Although neural networks have been applied to sensor fusion, their use in situation awareness has been limited, possibly because of the lack of rich training data for this problem." This is an important statement in the context of seeking a computational approach to SA. In particular, there have not been many attempts reported in literature to accomplish SA with a machine learning approach like neural networks. As the authors note, one reason for this is that the low amount of rich data is a problem for the learning algorithms. But this is also a general problem with many neural network implementations, i.e., the need for a lot of data.

Nevertheless, the authors cite the merit of using machine learning approaches. In doing so, they point to a fundamental reason for using machine learning in a computational approach to SA. "On the other hand, because they are data-driven, the advantage of machine learning techniques is that they can learn solutions to problems that are difficult for humans to codify with explicit rules or models. In other words, they can

represent rules/decisions that are implicit in the training data." This is a key point to emphasize: machine learning allows solutions to be developed that are not easily modeled. Furthermore, this is very much in line with what has been surveyed thus far in human factors. In many of the SA examples already mentioned, there was a learning process and decision-making loop that is not easily codified in terms of explicit rules. For instance, a pilot's SA in an emergency situation and his ensuing actions do not necessarily follow a prescribed rule set. This is a fundamental insight into the motivation of using machine learning for computational SA and it should be dually noted.

The authors bring data fusion principles into their implementation as well. They draw on the Hall & Llinas survey and observe that the analogy to animals fusing data from multiple senses is an inspiration for their SA implementation. "The analogy is helpful because fusion, and more generally situation assessment, is a process rather than simply a discrete event. The process leads one from raw data to understanding and actionable knowledge. Fusion can occur over various information (sensor) modalities, over geographic space, and over time." From this statement, it is quite clear that the authors understand the dynamic nature of SA. Furthermore, they explicitly describe the fusion problem as happening over both space and time. These are two elements that will be indispensable to the approach put forth in this work too.

An additional point on fusion is that the ways to do so are somewhat lagging behind the growth rate of sensors detecting data. This is somewhat akin to the information overload problem mentioned earlier in the context of NCW. The authors write [39], "Sensor capabilities in particular are maturing rapidly, but a valid concern is that the pace of sensor development has not necessarily been consistent with advances in human effectiveness which the sensors must ultimately support [.] Fusion algorithms will better support human-in-the-loop system effectiveness when the decision maker is a central and balanced design element." So the information overload problem is really a matter of communications to humans. Specifically, the fusion algorithms must

communicate to the human decision-maker in terms that make sense to him/her. Necessarily, this means an understanding of his/her goals. We see, therefore, how fusion and SA share common ground. In particular, data fusion would be useless unless it communicates to the decision-maker in terms they need to understand via their SA.

The authors go on to describe their computational implementation of SA in detail. Since machine learning requires data, the natural place to begin this description is with the data: "Key design attributes [to the situation assessment] ... include accepting various inputs such as binary, categorical, and real-valued data. With respect to situation assessment outputs, attributes include confidence levels as well as evidence in support or against the assessment." So we see that the information into this SA implementation spans a spectrum of formats. Furthermore, in line with our earlier survey of SA, the output consists of a situation assessment with some level of confidence. The evidence is an important addendum to the output because it provides a means for the human user to 'check' the accuracy of the assessment. In other words, the human user can assess the evidence on his/her own and then compare that assessment to that of the computational situation assessment. This is a worthy model of input/output to consider for computational approaches to SA.

The situation assessment is done with a module that receives both fused and raw data. At the heart of the data fusion is a neural network built from Adaptive Resonance Theory (ART) [40], which has been modified into an ARTMAP algorithm [41][42][43]. The extension from ART to ARTMAP is necessary because supervision data is explicitly input into the fusion module. The Coordinated ARTMAP (CARTMAP) is the name given to the fusion module used in this implementation. The step from ARTMAP to CARTMAP is where reinforcement learning is added to the module. Necessarily, this adds a significant level of sophistication to the SA implementation here. In some way, it makes the overall implementation comparable to a human learning about a complex system while receiving instruction from a teacher. Conversely, the removal of any *a*

priori knowledge (e.g., supervision or reinforcement data) would make such an implementation more comparable to a human learning about a system only from his/her own experience. It remains to be seen, however, which method is better. Necessarily, this will depend on the context in which it occurs, along with the other issues encountered in the complex systems survey.

But it is interesting to note in light of the earlier survey of data fusion how the data is fused. In particular, we should recall the second question from the Hall & Llinas survey: when in the processing flow should the data be fused? In the implementation done here, "the information fusion engine accepts raw data from sensors and other information sources and processes/transforms/fuses them into inputs appropriate for the Situation Awareness Assessment engine." But how raw is "raw"? This requires looking at the actual data:

> A dataset suitable for testing and demonstrating our technology was collected during a DARPA SensIT program ... The dataset consists of raw time-series (acoustic and seismic) and binary detection decisions from 23 sensor nodes distributed ... as one of two vehicles travels along a road. ... A scenario was developed whereby a facility under protection is assumed to exist along one of the roads, and binary sensor data processed by our fusion and situation assessment algorithms are used to inform a human decision maker.

Acoustic and seismic data is significantly raw in the context of force protection, which is the ultimate aim here. Other data used was passive infrared energy levels. Fortunately, the authors devise a processing structure that builds up the significance of the data to that which is most useful to a decision-maker concerned with force protection. So we see here how the information processing structure is very much determined by the type of data at hand. In particular, the fusion module needed to take in this data and output "vehicle

type, speed, location, and heading, each with a corresponding confidence level [that] will serve as input to the Situation Assessment module." We see, therefore, that the fusion happens on a low level in terms of the bottom-up processing structure. Specifically, we see that data fusion in this implementation plays no role in assessing the significance of the vehicle type, speed, etc. This is something worth noting because the approach to be developed later in this research posits that it may be possible to utilize fusion for higher-levels of situational understanding. Such a claim is made in the context of a given data source and not one as "raw" as the one used in the Brannon et al. implementation.

The latter stages of the information processing are less sophisticated than the fusion module. "[A situation] assessment formula was constructed/calculated, and a GUI was developed, all to increase the awareness of a human decision maker of the situation around that facility." The consequence of this information flow is that the "threat level is a function of ... the sensor array [data] and other variables that are independent of the sensor array." Consequently, from a high level, this information flow appears very much like those seen in complex systems research and SA formation/maintenance. The complete information flow can be seen in Figure 7.

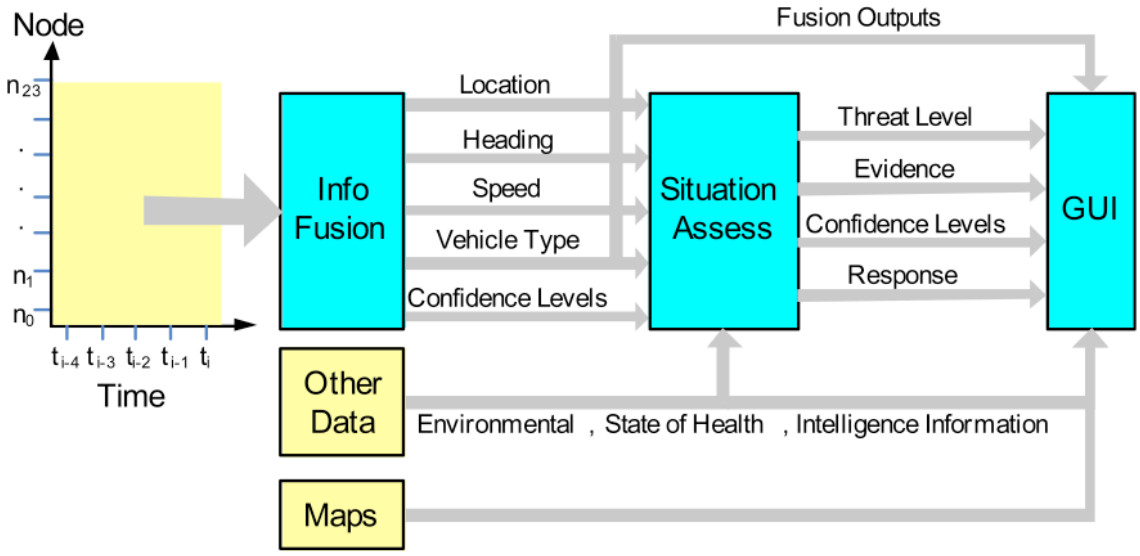


Figure 7: Information Flow for SA Implementation [38]

In addition to the already mentioned input/output to the fusion module, Figure 7 shows other sources of information used for the situation assessment. This information is available to the decision-maker in a graphical user interface (GUI).

Even though the fusion module operates from a sophisticated neural network, the situation assessment module is quite rudimentary. As the authors note, there are two implementations done here. The first one is less sophisticated than the second:

> A situation assessment module is performed by a weighted rule and a Bayesian filter. The weighted rule approach to situation assessment first transforms each input into a category ... [that maps each value of the inputs to a low/moderate/high threat level.] The next step is to compute the assessed threat level from a linear combination of all of the input categories, weighted according to their relative importance. ... The Threat Index is then converted to a threat category, to be presented to the decision maker.

This categorical mapping is necessarily heavily influenced via *a priori* knowledge. For example, the transformation of vehicle speed to corresponding threat level of that speed is an *ad autoritatem* processing task. Similarly, the weightings in the linear combination are also *ad autoritatem*.

The second approach has a degree of sophistication to it above that of the first. In particular, this approach relies on a Bayesian filter, similar to the kind used to detect spam emails [44]:

> The second approach to situation assessment designed for use in the force protection scenario involved a Bayesian filter. We generate reasonable estimates of the conditional probability, given expectations about the environment and

> interactions. ... The weighted rule formula used in the previous section can be used to establish initial conditional probabilities for the Bayesian Filter.

Even though this approach is more closely related to Bayesian inference methods, there is still a substantial role for *a priori* knowledge in the setting of the conditional probabilities. The authors note that this step, i.e. the actual situation assessment, is an area that needs further research.

From this approach to computational SA, we see how machine learning features so prominently. We also see that machine learning was used here for the fusion step. In particular, this step translated extremely raw data into higher-level data that was then categorized via *a priori* guidelines that corresponded to threat levels. This processing flow was evidently chosen because of the data at hand (i.e. the DARPA SensIT data). The situation assessment module was done either rudimentarily (the first approach) or with significant dependence on *a priori* knowledge (the second). Furthermore, it seems that machine learning played no role in the actual situation assessment module. Rather, machine learning allowed a fusion of data that was then mapped to previously known categories of threat level. While this is an interesting, and apparently somewhat successful, approach to take, the computational approach to SA posited later in this research attempts to blend the fusion step with the actual assessment step. As we shall see, this depends heavily on the dataset in hand when doing so. But first, let us survey briefly one final example of a computational approach to decision-making.

Case 3: Strategic Decision-Making Support

Kohl and Miikkulainen provide the last example to be considered in the surveyed lessons from the state-of-the-art in computational SA [45]. In this work, emphasis is placed on the computational aspects to decision-making, rather than SA *per se*. Also, this example does not specify NCW as the context. Rather, the authors consider some

benchmark problems in light of their approach. Their approach is an elaboration of neural network solutions called *neuroevolution*. In this approach, both a network's topology *and* weights are iteratively determined, rather than just the weights. Additionally, this work provides a mathematical description for why decision-making problems are difficult. While the exact formulation of the problem is distinct from the one to be encountered in the rest of this research, there are nonetheless some useful ideas to extract from "Evolving neural networks for strategic decision-making problems."

The authors begin with a brief survey of neuroevolution in the context of decision-making and they highlight one of the significant hindrances in its execution – a concept called *fracture*. The authors write, "[Problems of strategic decision-making] have remained difficult for neuroevolution to solve. This paper evaluates the hypothesis that such problems are difficult because they are fractured: The correct action varies discontinuously as the agent moves from state to state." We see from this that the authors assume that the state already exists. From the states, the agent (or decision-maker) must choose a course of action. Of course, knowing the states as inputs summarizes much of the data fusion and situation assessment done in the previous examples. In the context of a NCW military (Figure 3), the work of Kohl and Miikkulainen compare best to the execution part of the command & control (C^2). But these problems of execution, or decision-making, remain difficult and so the authors propose modified versions of an evolving neural network as the solution.

They propose such a sophisticated class of algorithms to handle the fracture of decision-making. The authors write the following on fracture:

> [Certain] types of problems - such as high-level decision tasks – still remain difficult for neuroevolution algorithms to solve. This paper presents the *fractured problem hypothesis* as a possible explanation for this issue. By definition, fractured problems have a highly discontinuous mapping between

states and optimal actions. As an agent moves from state to state, the best action that the agent can take changes frequently and abruptly.

Such an assessment of decision-making agrees somewhat with what was seen earlier in the human factors research. Recall, there we saw that decision-makers categorized a situation – something that has been assumed as input here – and from there chose a course of action that satisfied the goals. There is no continuous function that maps the category to the course of action. Likewise, the authors here see no continuity in that mapping. As a result, they have summarized such a problem mathematically in the concept they call fracture.

Since fracture is a mathematical concept, the authors attempt to define it as such. They use function variation to do so [46][47]. To use function variation though, one must have inputs and outputs. Consequently, "For this work, a problem is considered a 'black box' that already has associated states and actions. In other words, it is assumed that the definition of a problem includes a choice of inputs and outputs, and the goal of the agent is to learn given those constraints." What we see here therefore is a stripped-down version of the decision-making process. There is no knowledge discovery from the data at hand, as there was with data fusion or data mining. Rather, here machine learning is used to map states to their required actions. This therefore assumes a set of goals that have been constructed and implemented to get the outputs used here. They continue, "Because the variation calculation does not care what form the function takes – it only requires input and output pairs from the function – it is straightforward to calculate the variation of a neural network." So this is in fact why function variation is used to quantify the fracture of a decision-making problem. Though it is a useful concept to discuss fracture in the context of decision-making, it seems to be a rather strong assumption that states map to a prescribed set of actions. This leaves no room for novel uses of information in

decision-making. As we saw in the context of NCW, this is something that decision-making analyses must incorporate.

There are some other problems with the approach taken in this decision-making analysis. The authors write, "One simplifying assumption made in this paper [is] that there is a relatively smooth continuum in both score and fracture between poor policies and optimal policies." Thinking back to the human factors survey, there is cause to cringe at this statement. In particular, recall, "decision makers typically employ a *satisficing* strategy, not an optimizing procedure." Consequently, there is no such thing as an optimal decision – or "policy" – as the authors claim. However, the authors are somewhat following the line of thought posed in human factors research in that they assume a categorization of the state precedes action. They questionably represent this here though as a set of known inputs and outputs.

One other issue with this implementation is that it follows processing steps that are quite dissimilar from those believed to happen in actual decision making. The core of the authors' approach is an evolving neural network called Neuroevolution of Augmenting Topologies (NEAT). This neural network is designed "to solve difficult reinforcement learning problems by automatically evolving neural network topology to fit the complexity of the problem." This is done by genetically encoding the network structure, allowing for mutations to change both weights and structure. Also, "NEAT speciates the population so that individuals compete primarily within their own niches instead of within the population at large." Finally, "NEAT begins with a uniform population of simple networks with no hidden nodes. New structure is introduced incrementally as structural mutations occur ... In this manner, NEAT searches through a minimal number of weight dimensions and finds the appropriate level of complexity for the problem." There is no question that this is an impressive set of steps to find a suitable neural network for a given set of inputs/outputs. However, there is no indication from human factors research that this is what humans actually do when they make a decision.

Of course, there is some corroboration from complex systems research to employ such an approach: here, a complex system in the form of evolved neural networks is used to understand another complex system, i.e., the decision-making that happens from an observed set of situation states.

Ultimately, the authors offer two elaborations on the NEAT architecture to better handle some benchmark problems. However, it is not particularly clear how these problems relate to more complex decision-making tasks such as those that exist in the context of NCW. For instance, the authors use radial basis functions [48][49][50] and cascaded structures [51] to augment NEAT. They go on to conclude, "Both RBF-NEAT and Cascade-NEAT offer improved performance on all problems [surveyed here, such as generating maximal variation, function approximation, concentric spirals, multiplexer and keepaway soccer]," where each of these problems are fractured problems that serve as benchmarks [51][52][53]. But none of these contexts are comparable to the context of NCW decision-making, and so it is not clear how to use these results in that context.

On the whole, the Kohl and Miikkulainen work presents some interesting ideas about computationally implementing decision-making. But there are some conflicts with what was seen in the survey of human factors research. Furthermore, it is not clear how the strategic decision-making of their work relates to what happens in network-centric warfare. These are two points that must always be considered when computationally implementing an approach to decision-making. It will be necessary to keep them in mind as we proceed to the approach explored later in this research.

A New Approach to Machine Learning

Having surveyed some pertinent issues and applications in machine learning, data mining and data fusion, it is possible to reconcile these observations with what has also been surveyed in SA and complex systems literature. In fact, much of this commentary has been done along the way and we have seen the need for machine learning algorithms

to more closely emulate the information processing mechanisms of humans. This is necessary if computational SA is to become a reality. With good computational SA, good decision-making can happen, and that is the ultimate aim here. But, along the way, the lessons from complex systems must be kept in mind. In particular, we cannot take apart the phenomenon being observed to understand it, unless the interrelationships between the parts are first maintained. Also, observations of this phenomenon are the *sine qua non* for schema formation and so attention must be given to the type of data used. With these considerations in mind, a new approach to machine learning shall now be discussed.

This approach to machine learning is called Hierarchical Temporal Memory (HTM) [54][55][56]. It claims heritage from many avenues of machine learning and, most importantly for SA, from cortical information processing. Due to the newness of this research, there is a limited amount of literature available on it thus far. Yet, there is a growing number of applications appearing in the literature (e.g., for song identification [58], anomaly detection [59] and image classification [60]). Its theoretical foundations are encapsulated in two works by Dileep George: "How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition" [55] lays out a mathematical framework and "Towards a mathematical theory of cortical microcircuits" [54] describes how this approach matches anatomical data. We shall now consider aspects of these works to show why this approach seems a logical choice to explore in light of our preceding literature survey.

Overview of Mathematical Framework of HTM

Beginning with the mathematical framework, there are three major contributions from "How the Brain Might Work." The first is that learning and invariant recognition algorithms are developed for modeling hierarchical and temporal data. As our survey of human factors and complex systems showed, it is necessary to understand a complex system in time and with a suitable description provided by a schema. A hierarchical

approach to do this seems particularly attractive because it provides a generative mechanism for information condensation. Second, this work considers the generalization properties of these algorithms to other learning problems. Generalization properties will be useful to extend HTM to SA because the examples used in this work focus on invariant visual pattern recognition. Finally, this work overlaps significantly with "Towards a mathematical theory of cortical microcircuits" in that it presents much of the data about how the algorithms map to anatomical data. Though there are shortcomings identified in this mapping, it provides a significant step towards understanding human cognition from a computational perspective. As a result, this contribution makes Hierarchical Temporal Memory (HTM) attractive when considering how to accomplish SA computationally.

While all three contributions from this thesis are important, and will be considered in due time, the second contribution is worth considering now. This is because complex systems research and SA formation/maintenance, leading then to decision-making, are concerned with making suitable models for complex phenomena. So the generalization properties of these algorithms are necessarily worth considering. George writes, "Characterization of generalization in the HTM network is important because not all data domains and modeling techniques directly benefit from a hierarchical structure." He cites the fact that if nearest neighbor Euclidean distances can be used then there is no need to use a hierarchy. The data being learned must have a level of complexity to it that makes its mapping to a goal-oriented description not so trivial. Furthermore, "if there is no temporal structure in the data, application of an HTM to that data need not give any generalization advantage." This is because HTM depends on the slowness by which coincident events are assumed causally related.

It is important to consider this generalization characteristic in the context of what was said earlier about relational data. Specifically, the AFRL researchers in the first case of a computational SA framework said the following: "Relational learning allows

systems to exploit multiple tables in a database without the loss of information that occurs in a join or an aggregation. The resulting discoveries may include predictive patterns that more accurately describe the world by utilizing entities' attributes as well as the relationships between entities in the learning process." So even though the coincident events could be a false positive instance of causality, the learning of such a co-occurrence preserves information that otherwise would be lost by aggregation methods, such as defining rate-constants or probabilistic logic gates for simulations. The AFRL researchers continued, "While the challenges are significant, so too is the potential payoff." And so it is suspected that there would be a substantial payoff to using HTM for computational SA.

But the question remains as to why a hierarchy is necessary to model observed data. There is no direct answer to this question, although there is much research pointed at why it in fact makes sense. In this regard, George cites a well-known argument by Herbert Simon that hierarchical organization could be a general property of physical and biological systems [57]. In his work, Simon posited that learning machines would need to replicate and even to extend this structure. Similar arguments are also found in other literature for studying the natural world [61][62][63] and in systems science [64]. There is also substantial literature on the utility of hierarchy in machine learning techniques [65][66][67][68]. While some of these techniques bear relation to HTM, there are differences from how they incorporate the slowness of time to how the hierarchy is used to condense the learned/observed information. Each of these differences, ranging from *a priori* supervision (e.g., [67]) to the omission of spatial hierarchy (e.g., [69]) disqualify them from being suitable ways to model SA computationally because they remove at least one anatomically fundamental aspect to human cognition. By seeing how HTM maps to cortical anatomy, a working hypothesis can then be generated about how to create SA computationally. This hypothesis is that HTM can be used to do computational SA. So to see how HTM maps to anatomy we shall now consider "Towards a mathematical theory of cortical microcircuits."

Mapping of HTM to Cortical Anatomic Data

HTM is inspired by observations of cortical function. George writes, "[HTM] is a theory of the neocortex that postulates the neocortex builds a model of the world using a spatio-temporal hierarchy. ... [The] operation of the neocortex can be approximated by replicating a basic computational unit – called a node – in a tree structured hierarchy." For George and others, the importance of the hierarchy is acknowledged by seeing it as a mechanism to model data that pervades both time and space. Taking in observations in time and space, George writes, "The feed forward output of a node is represented in terms of the sequences that it has stored. ... The HTM hierarchy is organized in such a way that higher levels of the hierarchy represent larger amounts of space and longer durations of time." Here we see an intriguing overlap with how Endsley described SA. In particular, Endsley said that SA is formed by the integration of information that allows a projection into the near-term future. Similarly, an HTM node stores sequences of information to condense the information learned during observation. When this ability is replicated in a hierarchical structure, this allows an HTM network to do this over greater spaces and longer durations of time. George writes, "The states at the higher levels of the hierarchy vary at a slower rate compared to the lower levels. It is speculated that this kind of organization leads to efficient learning and generalization [.]" The implication from George's work is that this process is behind how humans learn and generalize observed information. If this is true then HTM may provide a way to implement SA computationally.

Of course, the idea that humans learn and generalize information like an HTM needs some anatomical correspondence. A thorough summary of learning and inference in HTM are provided in a later section, but it is useful here to introduce some basics. George writes, "The process of learning an HTM model for spatio-temporal data is the process of learning the coincidence patterns and Markov-chains in each node at every level of the hierarchy." As we will see later, coincidence patterns (*C*) are unique learned

patterns, and Markov-chains (G) are the most likely temporal sequence of these patterns. HTM models are called *generative* because they use the bits and pieces from C and G when observing a new piece of evidence to infer what the state is of the overall receptive field. This necessitates that both top-down and bottom-up processing occur when an HTM observes a new piece of evidence.

It is useful to consider a simple example to witness HTM processing. Below, Figure 8 shows a simple node that has completed learning. Both C and $G = \{g_1, g_2\}$ can be seen in part *a* of the figure. This node has learned 5 coincidences and 2 Markov-chains. In part *b* of the figure, we can see how both bottom-up and top-down processing happen when evidence (λ) is presented to the node. More of the details of this process will be revealed in a later section, but for now we can consider the 5 coincidences and the 2 Markov-chains to see how they map to cortical anatomy. A circuit representation of this C and G is shown below in Figure 9. Each of the Markov-chains is color-coded and we see how the 5 coincidence patterns are linked in the bottom-up processing. A similar figure can be found for the top-down processing in the referenced work. But the main feature to recognize from Figure 9 is that each column represents a coincidence. Furthermore, these columns are connected via the Markov-chains. Quite obviously, from this circuitry, we see the impact of Hebbian learning on HTM.

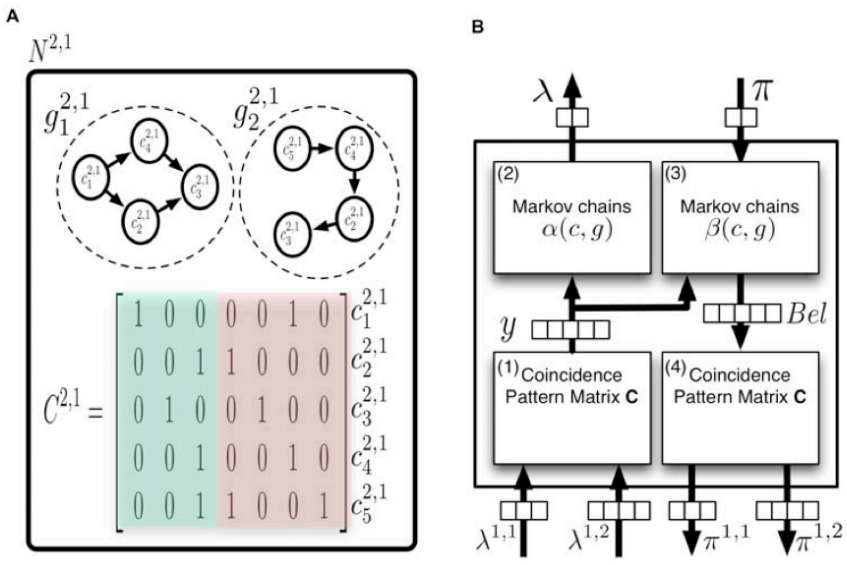


Figure 8: Simple Trained HTM Node [54]

The common expression "cells that wire together, fire together" clearly influences the formation of Markov chains shown in this figure.

But this circuitry can be mapped onto the cortex more directly. Figure 10 below shows the same circuitry mapped to the known six layers of cells in the cortex. There are a lot of details about this particular instantiation of a trained node to see here and it is all explained in the referenced work. Yet, there are some features worth pointing out here. First, the five coincidences that make up the columns of Figure 9 are represented in Figure 10 as five columns of neurons with feedback and feed-forward connections. The pyramidal cells of layer 2/3 explicitly represent each of the feed-forward coincidences seen in Figure 9. And, we can compare the green and blue circuits of Figure 9 with the blue and purple circuits of Figure 10 and see a similarity. These circuits represent the two Markov-chains and we can see that they are placed in layer 2/3.

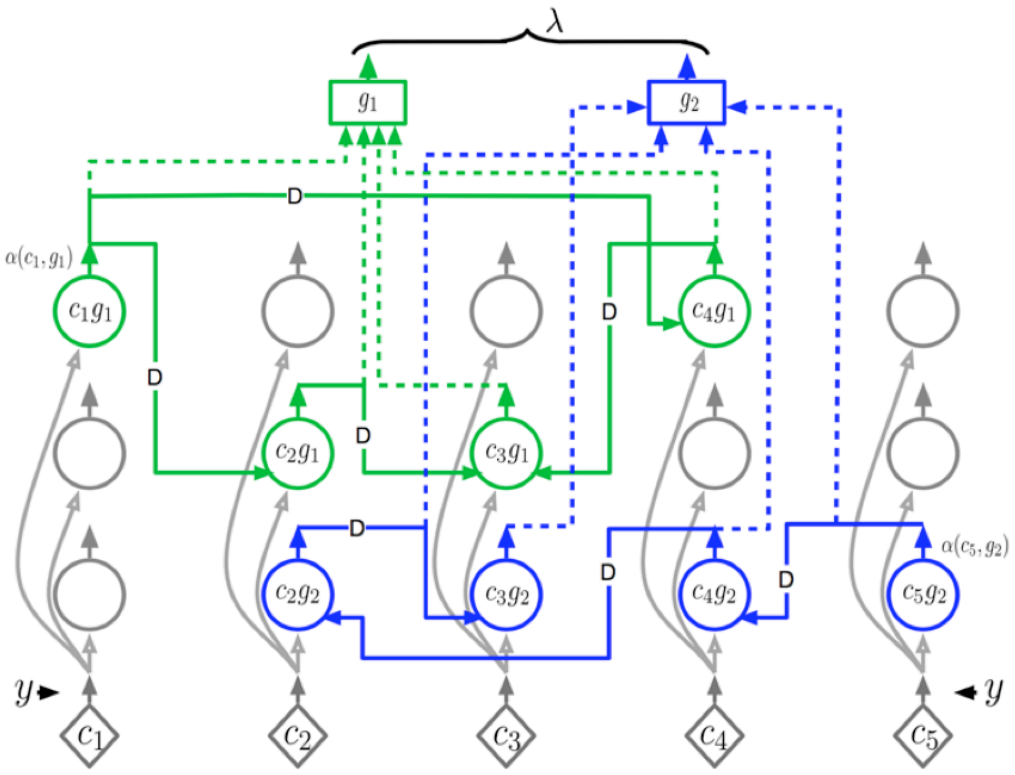


Figure 9: Simple Trained HTM Node Circuit [54]

George writes, "Cells in layer 2/3 are known to be 'complex' cells that respond to sequence of motion or cells that respond invariantly to different translations of the same feature. ... This is consistent with our proposal that most layer 2/3 cells represent different coincidence patterns in the context of different Markov chain sequences." Furthermore, top-down and bottom-up processing are both represented in this mapping to anatomy. George writes, "We show green and yellow layer 2/3 neurons in [the figure] because we need to learn two sets of sequences. One set of sequences is used in feed-forward calculations and the other set of sequences is used in feedback calculations. In our figures the green neurons are feed-forward and the yellow neurons feedback." So we see that HTM has built into it both bottom-up and top-down processing, two traits that have been noted to be important in the formation/maintenance of SA. But, as George writes, "This is a theoretical prediction currently without experimental data for support or falsification." In other words, there has been no direct anatomical observation that there

are neurons responsible for feed-forward processing and another set responsible for feedback with regards to sequences. Nevertheless, we know from our survey of SA that in the operation of complex systems, humans perform Level 3 SA in which a projection into the future is made.

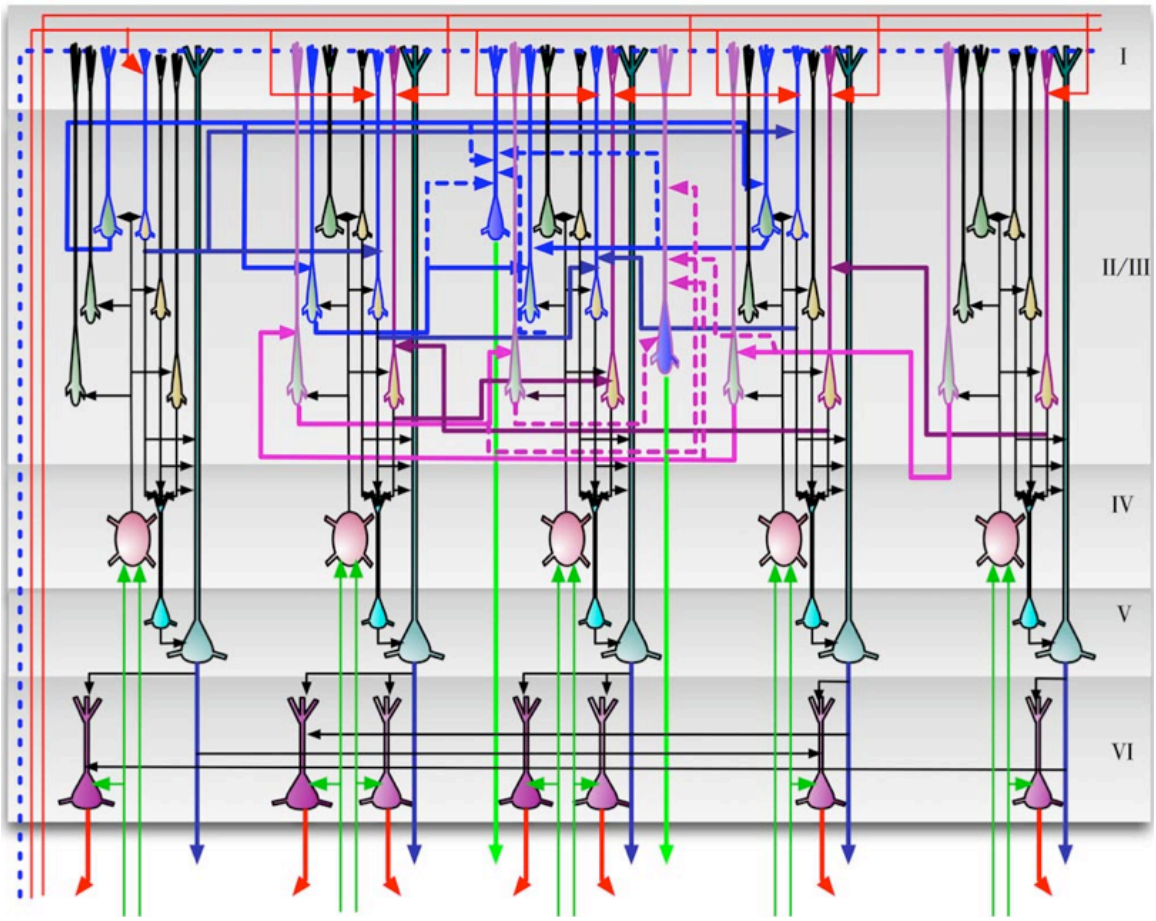


Figure 10: Simple Trained HTM Node Mapped to Cortical Hierarchy [54]

This is quite a similar concept to feedback processing of sequences of patterns and so it should not be considered to detract from HTM theory that no direct anatomical data has been found. Likely, the mechanism exists but not as simply as it is drawn in Figure 10.

Further mapping to anatomical data can be seen in layer 6. George writes [70], "Layer 6 is known to be a primary source of cortical feedback connections. ... The input connections to a layer 6 cell come from the columns corresponding to the coincidence

patterns that have the child [node's] Markov chain as a component." So we see that the black arrows coming down into this layer represent the feedback connections from each column, i.e., from each coincidence pattern.

Going on in this way, each layer of the cortical tissue can be mapped to different details of HTM computations.

Table 2: Mapping of HTM Computation to Cortical Anatomy

#	Anatomical feature	Proposed computational role
1	Feed-forward thalamic projection to layer 4.	Storage and detection of coincidence patterns.
2	Layer 4 cell dendrites are mostly within layer 4. These cells make vertical projections to layers 2 and 3.	Bottom-up inputs required for the sequence likelihood calculation in equation.
3	Layer 3 cells with inter-columnar lateral projections to other layer 2/3 cells. Some of these cells send their outputs to higher order cortex.	Calculation of sequence likelihoods for feed-forward and feedback calculations.
4	Layer 5 cells with apical dendrites in the superficial layer 4 and bottom of layer 3.	Belief calculation without specific timing.
5	Layer 5 cells with apical dendrites in layer 1. These send outputs to subcortical regions and non-specific thalamus.	Belief calculation with specific timing.
6	Layer 6 neurons with apical dendrites in layer 5.	Computation of feedback messages for child regions.
7	Projections to layer 1 from higher level regions and from non-specific thalamic cells.	High level input is feedback information. Non-specific thalamic input is timing information for Markov chains.

George summarizes these findings in Table 2. This mapping is of course a substantial simplification of the cortex. George writes, "The six-layered architecture we have described so far is most typical of sensory regions of cortex. Many variations in cortical architecture are known to exist [and they are] not explicitly addressed in our model." Some of this variation is person-to-person and some of it is generally true.

Nevertheless, the merit of HTM is that it can computationally execute many aspects of information processing that have been seen in SA research. Of course this mapping will not be perfect, just as George's mapping to the visual system in Figure 11 is not perfect. Here, we see that the six layers of Figure 10 are shown in part *B* as a slice from part *A*. When primary visual cortex (V1), secondary visual cortex (V2) and V4 are rearranged into a top-down and bottom-up arrangement in part *C*, we can see how both types of processing happen between sets of these layers.

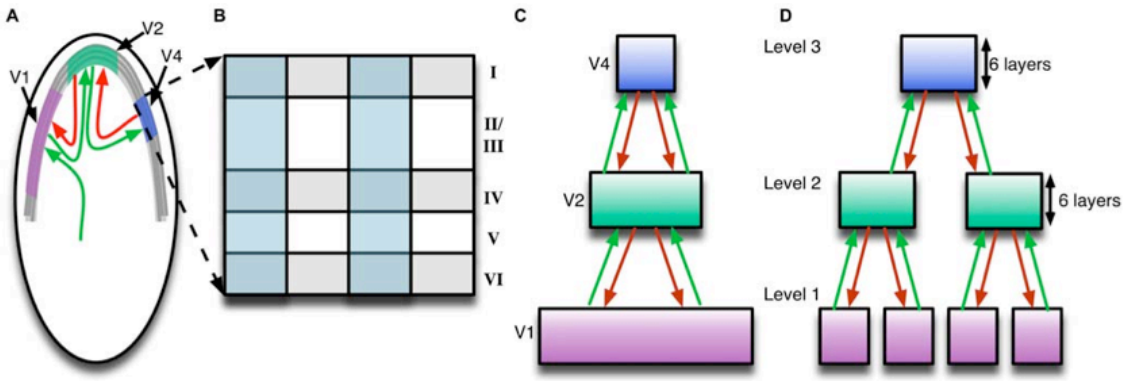


Figure 11: Mapping Between HTM Network and Visual System [54]

Part *D* then translates the anatomical hierarchy to that of an HTM. In part *D*, each level of nodes now contains the six layers of Figure 10. The real visual system is not arranged this simply at all. This can be seen if we compare part *C* of Figure 11 with an actual hierarchy observed from anatomical data (Figure 12).

Figure 12: Primate Visual System from Anatomy [71]

Of course, George has simplified the mapping dramatically, but as we will see later, HTM networks are remarkably well-versed at visual object pattern recognition. The question to be probed in the course of this research is whether a similar mapping can be made between HTM and SA. Visual object pattern recognition has been the context for much of the experimental work on HTM. In other words, HTM has been used for visual object SA. But would HTM be a suitable way to implement SA in another given context?

The Final Piece: Specify a Context

We see in HTM a potentially new way to do or at least augment situational awareness. However, it needs to be tested. Complex systems research tells us that we must specify a context to do so. What context can be chosen from the innumerable possibilities? Thus far we have restricted our focus to decision-making within the context of network-centric warfare. We will therefore continue to do so as we select a more specific context. Specifically, we want to focus on the situational awareness aspect to decision-making in this context. Because Endsley and others have told us that good SA leads to good decision-making, it makes logical sense to focus our efforts here.

What is the Context and Who is Analyzing It

We need to study a context that is both complex and requires decision-making sequentially in time. This context will be within the greater context of network-centric warfare so that our previously surveyed cases can serve as our benchmarks. With these thoughts in mind, along with current world events at the time of this research, the following context has been chosen: the U.S. Coalition Forces' attempts between May 2003 and April 2008 to provide stability to Iraq after the fall of Saddam Hussein. Specifically, we want to augment a high-level decision-maker's SA in the course of the conflict.

Just as it is important to specify the context, it is important to specify the decision-maker's role because this determines his/her goals. We know from our survey of SA that goals determine perception in a top-down fashion. So it is important to specify not only *what* is being perceived but also *who* is doing the perceiving. Now that we know what the 'what' is, we must also specify the 'who' in more detail. We have said a high-level decision-maker, so to whom does this refer? Is it the Secretary of Defense? Is it the President? Is it all of Congress?

The big problem we encounter when we start looking for a high-level decision-maker is that there is no one person that does this. Rather, there is a complex structure called the U.S. Government that serves the function of decision-maker. Thinking back to Gell-Mann, this is a similar situation to what exists in complex systems research: a complex system (United States Government) is used to understand another complex system (Second Iraq War). But, this analysis need not be done in a vacuum. Rather, as SA research shows us, a goal can guide the acquisition and maintenance of SA from the top down. In this case, there is a concrete goal that has been publicly declared by the U.S. Government in its aim to comprehend the Iraq Conflict: stability [72]-[83]. While the veracity of this declaration can be and is thoroughly debated, the fact remains that this is the declared goal, and so assessment of progress towards that goal hinges on data that either indicates or contradicts its attainment.

The Department of Defense (DoD) has been and remains the primary element of the U.S. Government that measures, assesses and operates to augment the stability of Iraq. Consequently, our focus on whose SA is being analyzed can be narrowed to this department. But there is no one person in mind here. Rather our inquiry focuses on what information can be synthesized to form an SA that would be of interest to the DoD as a whole. In other words, our focus is on the SA of the entire DoD. The DoD has declared its goal and its context has been specified; therefore, it should be possible for it to develop SA about the stability of Iraq. Since the DoD receives its directives from

Congress, we will approximate the DoD's SA as representative of that of the U.S. Government in its aim to stabilize Iraq.

Extracting Information from the Context

There has been much debate about how well or how poorly the DoD is accomplishing its declared goal of stability. Fortunately, the efficacy of its operations has been measured by a growing set of metrics [84][72]-[83]. As these metrics are measured and reported in time, they provide a description of the stability situation in Iraq. For example, consider the number of Iraqi civilian fatalities from May 2003 to March 2008 (Figure 13).

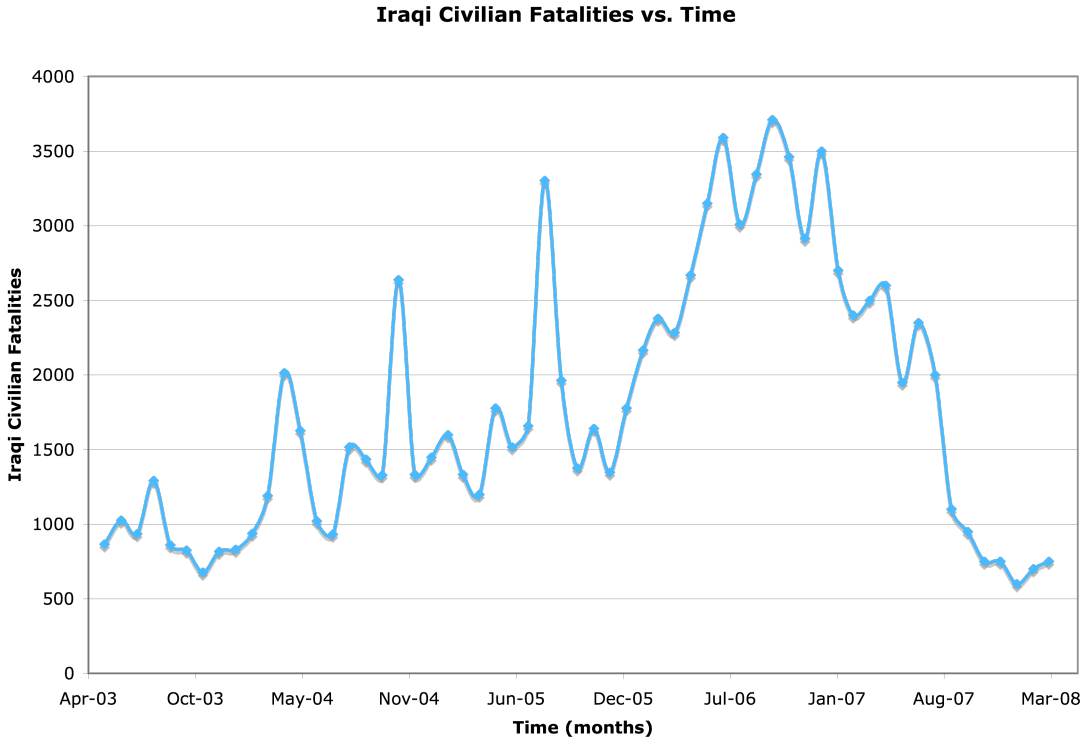


Figure 13: Iraqi Civilian Fatalities [84]

Assume for a moment that the number of Iraqi civilian fatalities equates to the level of instability in Iraq during this time period. If this is true then, given the DoD's goal and its perception of this relevant metric, we can conclude that the DoD SA would recognize the

spike over April 2004 as a more instable month than any measured month up to that date. Similar statements can be made about the preceding valleys leading up to the spikes at November 2004 and August 2005. Over the entire course of the observation, instability generally increased as civilian fatalities generally increased, until in January 2007 a general decrease in civilian fatalities indicates – per our assumption – a general decrease in instability.

But let us consider another metric used to measure stability (or instability) in Iraq during this same time period. Let us look at the nationwide unemployment rate. This metric is used by the DoD as a measure [72]-[83] of economic stability in Iraq.

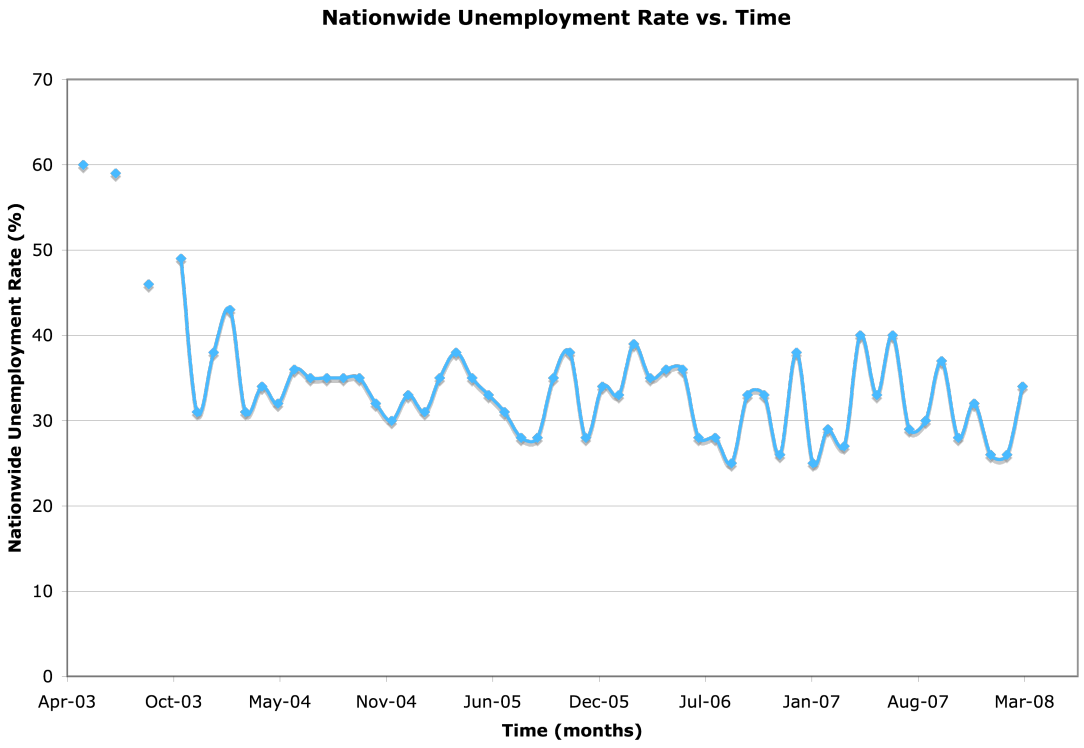


Figure 14: Nationwide Unemployment Rate [84]

Let us make a similar assumption to what was made earlier about Iraqi civilian fatalities. Specifically, let us assume that the nationwide unemployment rate equates to the level of instability in Iraq. Consequently, we conclude that Iraq had a decrease in instability between May 2003 and December 2003. From then on, Iraq maintained a generally

unchanging level of stability for the remainder of the time period. But if we compare this assessment based on nationwide unemployment with the one based on civilian fatalities then we come to two different conclusions about the same context.

Necessarily, as the quantity and diversity of these metrics grow, stability becomes an observable that is increasingly difficult to characterize. At one point in time, there can be both evidence for it and evidence against it. So in an evidence-based situation assessment there can be difficulties in coming to a sound conclusion. But we know from our survey of SA that decision-makers categorize a situation before proceeding to action. In other words, they come to some sort of conclusion about the evidence before proceeding with action. Considering DoD SA, it is logical to claim that the DoD also periodically comes to a conclusion also about its efficacy in stabilizing Iraq. From this conclusion, actions are taken or adjusted to steer the context towards a desired goal. But due to the breadth of descriptions (i.e., data) available about the stability of Iraq this becomes increasingly difficult. Can it be done better than it is done now?

So far, we have seen the important role evidence plays in schema formation, SA maintenance, the machine learning case studies and HTM. In each of these situations, evidence is recorded in time and condensed into a more storable form. In each situation, this process is different though. But can this evidence be condensed into meaningful information? The question of information's meaning has come up before in the context of NCW [8]: "[the] potential for information overload is real and great care must be taken to make sure that what is provided [to a decision-maker] is actually information and not noise." The stability of Iraq is a similar case, if not a subset case. Thus the question is how to form a better schema for Iraq's stability, i.e., how to give the DoD better SA on Iraq's stability.

Summary of Literature Survey

The preceding sections have surveyed a vast amount of literature across a broad array of disciplines. Complex systems research has told us that the analysis of a complex system requires a holistic perspective that does not isolate one element from any other. We saw some techniques for performing such analysis. They and many others come from machine learning, statistical physics and information theory. We then approached the problem of complex system analysis from the perspective of how humans do it. This required an exploration into human factors research. Endsley and others showed us that humans operate some complex systems by a multi-layered and evolving information-processing task that results in situational awareness (SA). We narrowed our focus particularly at times to the SA of individuals in network-centric warfare situations. This was not a more unique form of SA than any of the others discussed by Endsley, but complex systems research told us that the specification of a context is important. Consequently, this context received more attention than others. Seeing that some of the subtasks to SA had substantial overlap with machine learning and pattern recognition, we returned to possible computational implementations of SA. Specifically, we surveyed techniques of data fusion, data mining and machine learning, all of which had analogies in how humans form SA and what complex systems researchers wish to do in their analyses. Even more specific to our context of interest, we narrowed our focus to techniques proposed for use in combat or pre-combat situations that arise in network-centric warfare. We looked in particular at the use of machine learning techniques for decision-makers in such situations. Noting some useful contributions and some shortcomings in light of our earlier surveys, we narrowed the focus on HTM as a potential machine-learning tool. We showed how HTM maps well to anatomical data of the cortex and we have posited an idea from this survey that HTM could be of use for computationally implementing SA. This example case is the U.S. Department of Defense's SA in the context of Iraq's stability from April 2003 to May 2008. While we have no doubt that such an implementation will be rudimentary, it is suspected that such a

set of experiments would pave the way for a new paradigm in which decision-makers are aided by computational advice. So we now proceed to set down research questions devised from this literature survey and some corresponding hypotheses to set the way forward.

CHAPTER 2

RESEARCH QUESTIONS & HYPOTHESES

The preceding literature survey has left us with a perspective that spans many disciplines. From this, we are now able to synthesize some research questions. These research questions will motivate testable hypotheses. But the experimentation will be wrought with technical challenges. So their proper accounting will be needed to allow us to perform our experimentation with scientific rigor. With some of these challenges identified, a course can then be laid to research our problem in detail.

Research Questions

In the course of the preceding survey, there were many instances when questions arose. For example, how does one effectively analyze a complex system? How do humans analyze and operate certain complex systems? How can computers help us in this regard? These are some of the larger questions we have raised and answers proposed by the relevant literature have been reviewed. But in the course of following those arguments, additional questions have arisen. In particular, we have acknowledged the importance of situational awareness (SA) as a unique information-processing mechanism that is generally believed to allow humans both to comprehend and to operate some complex systems. Reconciling this insight with needs in network-centric warfare decision-making, we ask how to create better situational awareness for decision-making in this context. Adding in the growing amount of data available within successive decision-cycles, we ask how to do this faster and with more input data. As a result of our literature review, our research questions can be summarized as follows:

I. What are some improvements that can be made to the situational awareness of decision-makers in specific network-centric warfare contexts?
II. How can NCW situational awareness be acquired faster in light of there being more data to comprehend?

These questions are particularly high-level in terms of their scope. Moreover, the taxonomy of ensuing questions must follow to get to the specifics required to answer these questions. In doing so, we need to specify – per the complex systems influence – a context in which these questions can be probed. This requires us to focus on a particular network-centric warfare scenario to be a representative complex system.

The particular scenario to be probed will be the U.S. Government's attempts between May 2003 and April 2008 to provide stability to Iraq after the fall of Saddam Hussein. We have discussed in the literature survey why a focus on the Defense Department is needed. In particular, the DoD has specified a goal and coarse graining [72]-[83] by which to measure progress towards that goal. Endsley's research tells us that these are key ingredients to SA formation/maintenance. Conseqeuntly, the SA of the Department of Defense in the context of it assessing Iraq's stability can provide an experimentation platform.

The literature on computational SA also motivates such a choice for an experimentation platform. If we recall the literature survey, this is a similar scenario to that which was probed in the first case study of a computational approach to SA. In that scenario, indications of Iraqi incursion into neighboring countries before the First Gulf War were analyzed. The goal there was to identify Iraq's intentions with regards to its neighbors and the coarse graining was determined by the data of the 140 mentioned events [33]. For our chosen scenario, the goal is identifying the level of stability in Iraq and the coarse graining is determined by the DoD and Brookings literature [72]-[83][84].

Secondary Research Questions

With this context specified, some lower level questions can be asked. Let us look at the first research question (RQ I). Regarding improvements to decision-maker SA in NCW, it was clear from the literature on Iraq's stability that there is a wealth of data available. Yet there is not a clear way to fuse this data, thereby leveraging as much of it as possible. If we assume that the evolution of Iraq's stability is a representative complex system then we can infer two things from complex systems research [2]: "Don't take it apart [and] don't assume that only a few parameters are important [.]" Therefore, we cannot look *only* at Iraqi civilian fatalities to determine stability. And, we cannot look *only* at nationwide unemployment rate to determine stability. Rather, we have to consider these parameters together because one quantity is not more important than the other in Iraq's stability. As the number of parameters – or metrics, in DoD parlance – increases, it becomes more difficult to consider everything together. As we saw before, this is because there can be evidence for and against stability at the same instant in time. So we need to investigate what techniques can be used to resolve possibly conflicting pieces of evidence into a coherent overall model.

This is where data fusion, data mining and machine learning provide insight, since these disciplines offer ways to form a model from evidence. But, as our survey touched upon, there are many techniques available for doing this. One technique stood out from this survey for its intriguing overlap with SA information processing: Hierarchical Temporal Memory. Given our question about how to improve a decision-maker's SA in NCW, the question arises as to what extent HTM can augment SA in this context.

In the context of assessing Iraq's stability, it is important to probe HTM's generalization capabilities. As George reminds us, "not all data domains and modeling techniques directly benefit from a hierarchical structure." Consequently, we need to ask ourselves whether the DoD problem of assessing stability in Iraq can benefit from a hierarchical condensation of the knowledge. Or, would less sophisticated techniques

suffice. George also reminds us, "if there is no temporal structure in the data, application of an HTM to that data need not give any generalization advantage." We know from our literature survey on Iraq's stability during 2003-2008 that such data does exist, but we do not yet know whether enough of it can be coalesced into a temporally meaningful sequence of events. Furthermore, it is not clear how much data is enough to extract meaning from it in this context. Will the HTM extract useful meaning from two metrics, four metrics, or eight metrics?

Another aspect to HTM generalization worth considering is how to analyze its validity in this context. In other words, what tools can be used to analyze the SA supposedly generated by an HTM in the assessment of Iraq's stability? George has demonstrated HTM's generative modeling with invariant visual pattern recognition. It is simple to validate HTM's use in such a context because human visual systems easily recognize and agree upon the identity of invariant visual patterns (in general). But such a luxury does not exist in recognizing levels of stability. Take for instance two perspectives on stability from a 2006 United Nations resolution to extend the term of the multi-national force [85]:

- "[The representative of the Russian Federation said the] situation in Iraq remained complex, he continued, and the signs of improvement were not evident."
- "[The representative of the United Kingdom] added that the multinational force had already been able to hand over control of two provinces to Iraq during this summer, and conditions permitting, he looked forward to notable progress in the next year."

Here we have two statements about the same situation from a UN Resolution. They offer two different perspectives on the then current state of Iraq's stability. In human factors

parlance, we have two different situation assessments as generated by two different persons' situational awareness. Necessarily, different goals and different perception of facts combine to make this happen. Conversely, with invariant visual pattern recognition, the goal and perception of visual facts do not change from person to person. The goal is to identify the image. The visual facts are contained within the boundaries of the image and are available for perception. But stability analysis in the context of Iraq is different. The assessment of stability is not confined this way. Rather, both the bottom-up discovery and the top-down goal-driven perception of certain facts determine the assessment of stability. So we must be absolutely clear as to which facts are being considered and which goals are driving the perception of these facts. These questions and issues must all be considered when analyzing the application of HTM to decision-maker-level SA in the context of Iraq stability.

Considering these issues and questions, the taxonomy of secondary research questions for RQ I is summarized as follows:

i. Can we leverage the available data simultaneously, rather than looking only at likely conflicting indicators?
ii. Would a hierarchical condensation of the data help to leverage the available data together?
iii. For a given context, can a comprehensive enough time series of data be collected and analyzed to extract meaningful information?
iv. What tools can be used to analyze SA in the context of stability assessment, especially when the goals and consequent perception of facts are subject to change?

These questions will serve as a guide when constructing testable hypotheses to be proved in the remainder of this research.

Now we must probe the second research question (RQ II) as we did the first. Fortunately, less exposition is required before doing so. The question is how to do NCW SA faster in light of there being an increasing amount of data available for leveraging a solution. Let us recall the motivation to do this from earlier NCW literature [8]: "[the] potential for information overload is real and great care must be taken to make sure that what is provided [to a decision-maker] is actually information and not noise." In a more general context, Endsley tells us, "In complex and dynamic environments, attention demands resulting from information overload, complex decision making, and multiple tasks quickly exceed a person's limited attention capacity." So how can a lot of data be operated on quickly? More importantly, how can these operations be done to yield meaningful insights into a given context? Per the example from the UN Resolution and many possible others [86][87][88], we wish these operations to be comparable in light of declared goals.

Because of the need for speed in processing, it seems that a computational solution would help, but what kind of solution? From our literature survey of machine learning, we found, "the advantage of machine learning techniques is that they can learn solutions to problems that are difficult for humans to codify with explicit rules or models. In other words, they can represent rules/decisions that are implicit in the training data." So it seems that some machine-learning algorithms can be of use, but it is not clear which ones. For instance, HTM seems to be a candidate because of both its computational implementation and its similarity to certain aspects of SA information processing. But as with all machine-learning solutions, no algorithm is perfect for all learning problems [89]. Rather, all learning algorithms make assumptions that are leveraged on the data to learn its features. Since these assumptions tie into goals, the question arises as to what these assumptions are and how the predetermined goals of our analysis affect them. Moreover, we wish to know how these assumptions affect the resulting SA they are meant to generate.

Taking these considerations into mind, the secondary research questions for the second primary research question (RQ II) are as follows:

i. Having specified a context and goals, how can we balance the need to computationally fuse lots of data with the need to yield meaningful results?
ii. How can we create a basis for comparison in our assessments?
iii. If machine-learning techniques are used then how will specific algorithmic assumptions affect the generated SA?

As with their counterparts earlier, these questions will also serve as motivational points to the hypotheses.

Hypotheses

In the course of coalescing our narrative into research questions, some lower-level questions have been raised that allow us to make testable hypotheses. Recombining these hypotheses according to the taxonomy of the research questions, we can reconstruct some higher-level hypotheses that combine to give us the primary research objective. This process of hypothesis generation is shown in Figure 15. So let us proceed in this manner to address our derived research questions.

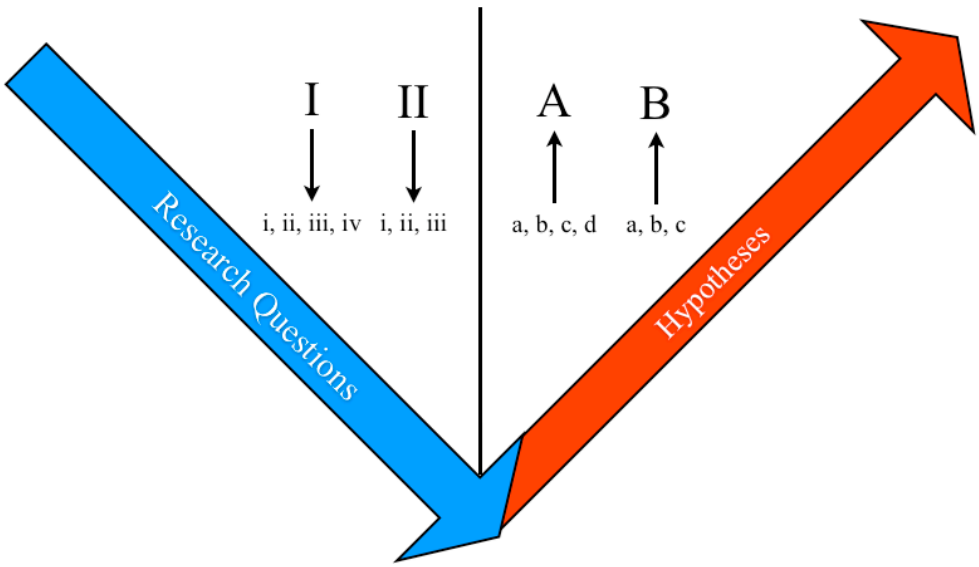


Figure 15: Research Question and Hypothesis Taxonomy

Secondary Hypotheses

The secondary questions of the first research question (RQ I) will first be answered with corresponding hypotheses. A brief reasoning for each hypothesis will then follow before the next question is answered. Let us proceed then to answer RQ I.i:

a. For a given context, the available data can be leveraged simultaneously when it is condensed into a schema that is suitable to the specified goals.

Since we wish to incorporate the data simultaneously, we draw upon insights from the SA of complex systems. Specifically, the hypothesis then is that it is possible to create a schema so that the data can be condensed, but only when goals have been specified *a priori*. Otherwise, the top-down search for relevant data is unclear. Looking at RQ I.ii, we propose the following hypothesis:

b. A hierarchical condensation of the data is possible, but the significance of it remains to be seen.

Since complex systems are made of interdependent parts, it is believed that a structure exists from which behavior can be learned. But the description of this structure is necessarily long, per earlier discussions on complexity. So a hierarchical condensation of information observed about the system might provide a generative mechanism for describing a given complex system. But the significance of this condensation in light of the goals that have motivated analysis remains to be seen. Looking at RQ I.iii, we propose the following hypothesis:

c. For a specified context and goals, a temporally structured dataset can be prepared and analyzed to extract high-level states evident in the data, although the meaning of these states remains to be tested.

Citing the need to specify a context and goals, the data pertinent to the DoD's assessment of Iraq's stability will be the chief source of information on Iraq's stability. Because this data comes in many forms, it is necessary to prepare the data in such a way that simulates the flow of events as they happened during 2003-2008. Other cases will also be examined leading up to Iraq stability assessment. In all of these cases, a context and a set of goals will be specified to guide the dataset's analysis. Furthermore, various operations can be done to condense this data into a numerical representation of the overall state implicit in it over some time interval. The meanings of these representations though remain to be seen. Also, they will likely differ, based on the types of operations and their assumptions about the system's dynamics. Finally, considering RQ I.iv, the following hypothesis is made:

d. To analyze an implementation of SA in the complex task of stability assessment, at least two approaches are needed: expert validation and extreme-case bounding.

Since it is not easy to agree on what constitutes a stable Iraq, extreme-case bounding will employ a system dynamics model to generate data that leads to asymptotically stable and unstable states of Iraq. This will require a predefined goal because that will in turn specify the relevant data to be monitored. Using the boundaries provided by this experimentation, the task is to determine the shades of stability in between. This will require a comparison between calculated stability state and expert opinion.

Proceeding to the second research question (RQ II), its secondary research questions shall now be answered with hypotheses as well. In answer to RQ II.i, the following hypothesis is made:

a. Specifying a context and a set of goals, an approach based on Hierarchical Temporal Memory can balance the need to fuse data with that of extracting useful meaning from it.

In the course of the literature survey, there has been constant emphasis on context and goals. If we wish to use HTM to extract useful meaning from data, then it will be necessary to examine this claim in different contexts in which there will likely be different goals. Nevertheless, if the generalizing principles for using HTM in other arenas are adhered to, then HTM should contribute goal-driven knowledge to the understanding of a given context. In response to RQ II.ii, the following hypothesis is made:

b. In assessment tasks of the same context, the goals determine the basis for comparison because these in turn determine the relevant observables.

The importance of goals in assessment of data cannot be emphasized enough. There is interplay between top-down and bottom-up information processing in assessment. In response to RQ II.iii, the following hypothesis is made:

c. For a given context, different machine learning algorithms – and even different implementations of the same algorithm – will likely create different SA in light of prescribed goals, but some techniques will yield more significant results than others.

As the literature survey on machine learning, data mining and data fusion tells us, there are many ways to extract information from data. The focus of this research is to use HTM to create a computational implementation of SA because of its argued merits over other techniques. The comparative analysis to other machine learning techniques will be minimal here in comparison to what is available in the relevant literature. Such work is beyond the scope of the prescribed research objective. Nevertheless, extensive testing of the sensitivity of various HTM implementations is certainly within our scope of study.

Reconstructing Secondary Hypotheses into Primary Hypotheses

Having hypothesized answers to secondary research questions, it is possible to ascend back up the red arrow of Figure 15 to the primary hypotheses. To answer RQ I, let us summarize the hypotheses made in response to the secondary research questions:

a. For a given context, the available data can be leveraged simultaneously when it is condensed into a schema that is suitable to the specified goals.
b. A hierarchical condensation of the data is possible, but the significance of it remains to be seen.

c. For a specified context and goals, a temporally structured dataset can be prepared and analyzed to extract high-level states evident in the data, although the meaning of these states remains to be tested.
d. To analyze an implementation of SA in the complex task of stability assessment, at least two approaches are needed: expert validation and extreme-case bounding.

We can now combine these hypotheses in order to answer RQ I:

A. SA of decision-makers in NCW contexts possibly can be improved with computational aides that exploit the hierarchical and temporal structure of the data relevant to a given context and prescribed goal.

We see that the secondary hypotheses are wrapped into hypothesis A (H A) either directly or indirectly. For instance, H A specializes down to H A.a because a computational aid can be used to create a schema that exploits hierarchical and temporal structure of data pertaining to a given context and goal. This assumption of the data's structure is crucial to its condensation into a schema and will necessarily affect its significance in light of decision-maker goals. H A.b even more clearly folds into H A. The results of tests for H A.c and H A.d will have the most affect on conclusions about the validity of H A. This is because the majority of the analysis required to validate H A is contained in these two secondary hypotheses.

We can ascend the red arrow of hypotheses to answer RQ II as well. The secondary level hypotheses are summarized below from their earlier discussion:

a. Specifying a context and a set of goals, an approach based on Hierarchical Temporal Memory can balance the need to fuse data with that of extracting useful meaning from it.
b. In assessment tasks of the same context, the goals determine the basis for comparison because these in turn determine the relevant observables.
c. For a given context, different machine learning algorithms – and even different implementations of the same algorithm – will likely create different SA in light of prescribed goals, but some techniques will yield more significant results than others.

As before, these secondary hypotheses can be synthesized into a primary hypothesis that responds to RQ II:

B. It is likely that NCW SA can be formed quickly and maintained usefully through schema formation based on Hierarchical Temporal Memory.

As with H A, hypothesis B (H B) has secondary hypotheses H B.a through H B.c wrapped into it. For instance H B.a, claims that SA based on HTM will be of use in contexts where goals have been specified and relevant data is available. In the course of testing this claim, the speed of HTM's SA formation can be probed. The other contexts and goals in which HTM-based SA shall be formed will then serve as preceding examples to lead up to the NCW SA example of Iraq stability. Similarly, H B.b and H B.c fold into H B.

Now that there are two primary hypotheses, we see that these hypotheses combine to satisfy the research objective established from the beginning. This research objective is encapsulated in the title to this work, i.e., a computational approach to the situational awareness of complex systems. Necessarily, a specific context has been chosen when

proceeding from the general research objective to the primary hypotheses. But this shall serve, with intermediate examples along the way, to illustrate a computational approach to the situational awareness of complex systems.

Technical Challenges

The taxonomy of research questions and hypotheses leads us to technical challenges that must be encountered as we proceed to experimentation. These technical challenges particularly focus on the implementation of machine learning techniques to be used ultimately for NCW SA. There are various recommendations from relevant literature about suggested processes to follow when computationally implementing SA. As a result, choices will be necessary depending on the context and goals in question.

These technical challenges can more concretely be formulated by looking at each of the secondary hypotheses to be tested by experimentation. For instance, H A.a requires us to choose how the data is fused into a schema or set of schemata. It is both challenging to pick a data fusion process and to determine how the data should be fused. Before fusing it, it must be prepared in a particular way as well [90]. H A.b provides a challenge in mapping the hierarchical condensation of data to more qualitative significance. Similarly, for H A.c, there is a challenge in preparing the data properly and assessing once again the qualitative significance of the determined high-level states. A lot of how H A.b and H A.c are tested will come from how H A.d is performed. H A.d will require a similar mapping between qualitative and quantitative results, but it will also require something else. Specifically, H A.d will require a way to generate data about a complex system. Fortunately, the task is specific enough that a rudimentary modeling approach based on system dynamics can accomplish this. Nevertheless, it will be a challenge to do this in a way that provides suitable bounds for quantitatively analyzing stability assessment.

A similar set of challenges will arise when testing the secondary hypotheses to H B. H B.a will require extensive knowledge of HTM theory and implementation practices. H B.b is less of a directly technically challenging hypothesis, but experimentation to test it will be built on the other hypotheses' testing. Finally H B.c will also require extensive knowledge of HTM theory and implementation practices. Furthermore, a rudimentary understanding of the technical knowledge behind other machine learning techniques will be necessary to test this hypothesis as well.

In summary, the technical challenges will straddle two fundamental issues: how to implement machine learning training and testing, and how to interpret the results. With these technical challenges, we have defined the problem that is to be examined in this research. Now, to do so in more detail, a research plan will be established in the next chapter to see what solutions can be created for this problem.

CHAPTER 3

RESEARCH PLAN

The taxonomy of research questions has defined the problem we wish to examine. Similarly, the taxonomy of hypotheses has provided us with statements whose validity can be assessed with experimentation. The technical challenges then begin to show what issues will arise in the course of testing these hypotheses. So now it is necessary to formulate a research plan that will allow us to probe these issues in detail. From a general perspective, we want to find a way to execute the question-marked arrow below.

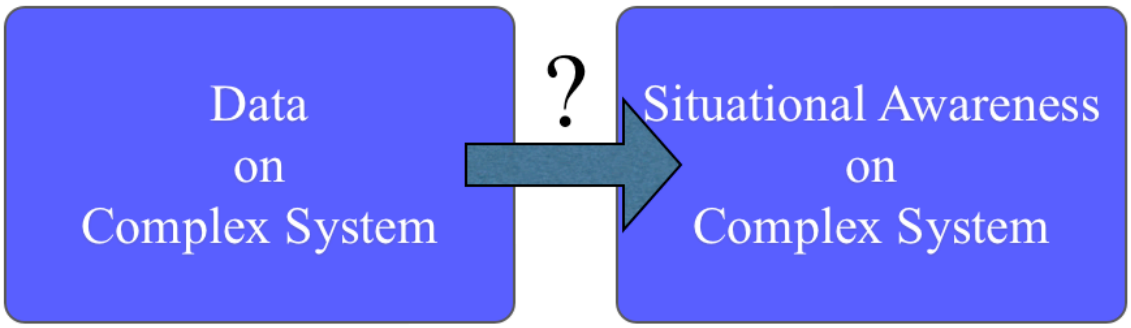


Figure 16: General Question Addressed with Research Plan

Thus the aim of the research plan is to figure out how what replaces the question-marked arrow in Figure 16.

Preliminary Considerations

Before laying out a research plan, there are preliminary considerations derived from the technical challenges that must be considered in greater depth. The first concerns how to computationally implement schema formation. There has been much emphasis thus far on the role of goals and contexts in the formation of SA. The context specifies

the available data and the goal specifies the importance of a subset of that data. Of course, the aim in forming SA in a context is to condense the perceived data into a schema that is useful in light of goals. One of the secondary hypotheses has proposed a hierarchical condensation of the knowledge (H A.b) to do this. Another secondary hypothesis has proposed the preprocessing of the data into a temporally structured course of events to ease condensation (H A.c). Yet still another secondary hypothesis has built upon these two to propose the possibility of doing this with Hierarchical Temporal Memory (H B.a). But how would this be done? Specifically, what would a hierarchical condensation of the knowledge look like? How could we tell that the knowledge was condensed this way? Regarding temporal structure of the data, what steps are necessary to do this? How do we do this in light of data not being available *a priori* this way? If HTM is to be used for schema condensation, then what would be the details of computational implementation? Would we start from known solutions to previous problems or attempt an optimization problem to find the optimal HTM for a given SA task? These are all questions that the research plan should answer.

The answers to many of these questions hinge on the nature of the approach that is used. Given the emphasis thus far on HTM, we shall propose ways to answer these questions from the perspective of an HTM approach. We shall still consider other methods but not nearly as in-depth. Let us choose as a working hypothesis then that HTM is a way to condense information about a given context into a schema. Necessarily, this requires specification of the goal that has guided perception of the data used for schema formation. Furthermore, the information about the context must be arranged into a standard form that allows HTM generalization. A procedure must therefore be established to do this. Some insight on how to do this will come from the preceding literature survey on data fusion and machine learning, though other insights from data mining will help as well. If HTM is to be used for schema formation then the available choices in designing such a network are limitless. Yet, there is guidance available from past implementations

that can help. Coming from neural network literature, one idea would be to employ optimization techniques to find the optimal parameter and topology settings of an HTM. For instance, this is how back-propagation works to find the correct weightings in a neural network. But there is a problem with this approach because we do not know what the 'correct' answer is. In other words, we cannot optimize when there is no objective function. If we are using HTM to find implicit knowledge that is "difficult for humans to codify with explicit rules or models" [38] then defining this objective function becomes difficult. Consequently, rather than embarking on an optimization problem, we propose an evolutionary approach from known HTM solutions.

If an evolutionary approach to HTM design is to be done, instead of an optimization-based approach, then we must find a set of HTM problem-solutions that can be modified to meet our goal of implementing SA. This requires them to be rethought as conduits to our primary goal of facilitating NCW SA computationally. In doing so, a theoretical framework will be needed to compare past approaches to ones developed here. For instance, the context and the goals in each approach must be specified (H B.b). Also, a mathematical framework would be useful to gain an understanding about how different problem-solutions are derived from general principles. These theoretical considerations would then have to be combined with the practical needs of implementing SA in a given context. All of these issues are important for devising a suitable research plan.

HTM is surely not the only solution to computational SA augmentation. As the literature survey showed, there are other possibilities coming from machine learning and data fusion. So the research plan must allow room for comparative analysis between HTM and other techniques of computational SA. Necessarily, a comprehensive survey of all possible techniques is considerably beyond our scope of interest. But some rudimentary implementations can be compared to see why this problem is not so trivially solved. This is where observations about run time would be appropriate because this would put HTM on a comparative footing with other methods in yet another regard.

Forming the Plan

It now remains to form these considerations into a logical plan of execution. Starting from our working hypothesis that HTM can be of use to answer our research questions, we must first dive into the theory behind HTM. The literature survey touched on the surface of how HTM exploits hierarchical structure and the temporal sequence of perceived events [54][55]. But this must be developed further. More importantly, George's theory must be considered in the context of our secondary hypotheses. We will have to find which generalities are already included in his formulation and to determine what further development is possible. The first step of this research plan can be stated as follows:

1. Explore the theoretical foundations of HTM, focusing on the needs of SA information processing.

By focusing on SA information processing, we will be able to see how past work on HTM can be conceptualized in this context. In particular, this would allow us to see how both context and goals are central pieces to HTM information processing.

Building on this theoretical framework, it would be necessary to consider past approaches pertinent to our primary objective of complex system analysis. This would require examples that lead the way to our primary complex system example: the Iraq context. But the road to this context will be long because of the rudimentary nature of the past approaches that will lead there. So it will be necessary to identify the lessons that can be extended forward to the Iraq context and where the gaps remain. Some of the tools encountered in our literature survey will help us to do so. Specifically, complex systems' specification of coarse graining will determine the comparability of an example to another complex system. Also, data fusion tools will be of service to see how HTM information processing fits into a larger context of human-computer interaction. Finally,

data mining practices will help us see where capability gaps exist when extending HTM to complex system analysis. As a result, the second step of the research plan is as follows:

2. Examine HTM implementations that are potentially pertinent to complex system analysis and identify where capability gaps exist.

This step will begin the path that leads to complex system analysis with HTM, but it will not be traversed without concrete examples leading the way.

This next step of the research plan must find a representative conduit example that can illustrate how to traverse the gap between what has been done and what is needed for computational SA. To do so, a representative complex system will be studied on a familiar level of coarse graining. This example is chosen not only for its familiarity to the aerospace engineering community but also for its ability to be reconceived in a complex systems perspective. This example is the analysis of gas dynamics in the presence of a disturbing supersonic body. The level of coarse graining most pertinent to the aerospace community deals with measurable quantities such as temperature and pressure, etc. So this level of coarse graining shall be assumed. Furthermore, the context needs to be refined to a set of scenarios readily tractable with traditional analytic tools. Consequently, the context of a finite number of supersonic bodies passing through a point in space is chosen. These bodies are further assumed to cause normal shocks in the gas, which is assumed to be equilibrium air for data-generation purposes. Having set this context (henceforth called the 'shockwaves context'), we will use this example to show how goals can influence the resulting analysis when done by HTM. Other computational methods of SA can also be examined here. By doing this, it is believed that this example will serve as a first canonical example that can lead to analysis of the Iraq context. So the third step in the research plan is as follows:

3. Analyze a familiar canonical problem that can begin to bridge the gap between potentially pertinent HTM implementations and complex system analysis.

Completion of this step will bring us closer to our goal of computational SA in the Iraq context.

But there will be notable differences between the problem analyzed in the third research step and that of the Iraq context. These differences will be noted from a system dynamics perspective. In the course of doing so, a strategy will then be devised to use system dynamics modeling to help bridge the gap. Specifically, a system dynamics model based on what is known about the Iraq context will be created to provide us with fictitious data on extreme cases of both stability and instability. Ultimately, this data will play a role in the training and testing of an HTM derived from previous work. However, the detailed nature of this focus on system dynamics requires it to be its own step in the research plan. So the fourth step is as follows:

4. Analyze both the canonical problem of step 3 and the Iraq context from a system dynamics perspective to lead the way to extreme-case bounding for stability analysis.

This step provides a bridge between the Iraq context and the shockwaves context, which was in turn a bridge to pertinent past HTM implementations. By introducing this system dynamics perspective, we also will retroactively be able to see how previous implementations were simplified versions of both the canonical problem and the Iraq context.

Having laid the foundation, the next step is to analyze Iraq's stability with HTM. At least two training/testing approaches will be done here. One approach will use the extreme cases as training data and use the actual data as testing data. The other approach

will use the actual data for training and the extreme cases for testing. The utility of both approaches will be assessed in light of both the quantitative evidence and the qualitative reports of the Department of Defense [72]-[83]. Regardless of the approach used, tests will be performed and reported on (1) alternate ways of presenting the data to the HTM, (2) alternate ways of choosing network topology and parameter values, (3) reducing the number of metrics used for training and testing, (4) examining the hierarchical condensation of Iraq context knowledge in HTM, and (5) examining the level to which HTM can extract implicit predictive information from evidence. Also, as a comparative reference, some rudimentary techniques for computational SA will also be presented and discussed. So the fifth step of the research plan is as follows:

5. Analyze the Iraq context with HTM and perform tests to improve our understanding of its utility in SA formation/maintenance.

The tests done in this step are certainly not meant to be exhaustive. Nonetheless, they can certainly augment our analysis to determine whether HTM is a useful tool for SA in this context.

Having assessed the utility of HTM in SA augmentation, we can then return to our illustrative complex system task. Specifically, this concerns a decision-maker's SA in the context of assessing and operating on Iraq's stability. This step will allow us to see the potential uses of an HTM approach to SA in the Iraq context. Furthermore, this might then allow us to judge the utility of HTM in the general task of complex system analysis. Therefore, the sixth step of the research plan is as follows:

6. Analyze the utility of the Iraq stability SA HTM in the context of decision-making and show how this fits into the larger context of complex systems analysis.

Necessarily, the discussion on the larger implications to complex systems analysis will be specific to the contexts explored in this research plan. However, in being so, it will be possible to identify those points necessary to consider in moving forward from this research.

Once this step has been completed, it is possible to return to the hypotheses in light of the accomplished experimentation. Consequently, the last step of the research plan is as follows:

7. Revisit the hypotheses in light of the accomplished experiments.

This research plan has been designed to test both the secondary and the primary hypotheses of the previous section. Once revisiting these hypotheses in light of accomplished experimentation, the primary research objective of this work can be revisited as well. Specifically, we shall then be able to assess our computational approach to the situational awareness of a complex system. Since this research focuses on example complex systems, it will not be possible to generalize to all complex systems. However, as stated earlier, it is hoped that the work done here can provide a foundation for future study of complex systems analysis. Some of these possible directions will be discussed in concluding remarks.

CHAPTER 4

METHOD FORMULATION

The first step of the research plan requires us to investigate HTM theory in more detail. The primary focus here will be completely dependent on the work of George [54][55]. Recall from the literature survey, George made three large contributions in the theory of HTM. We will consider the first two now. The first contribution concerns the development of learning and invariant recognition algorithms. The second concerns the generalization properties of these algorithms to other problems. In considering these contributions more in-depth, we will see the details of the learning algorithms as they construct a knowledge base. We will also see more in detail how the evidence-based inference processing functions in HTM. With these details, it will be possible to deepen our understanding about the generalizing capabilities of HTM. Then, we will be able to further probe the connection between HTM and situational awareness (SA) formation/maintenance. Having done this, it will be possible to see exactly to what extent HTM can be of use for a computational approach to SA. In doing so, we can utilize our literature survey to further develop the theory behind forming/maintaining a computational SA. This chapter then will cover steps 1 and 2 of the research plan. We now begin with the current state of the theory as presented by George.

Theory of Hierarchical Temporal Memory

In "How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition," George lays the foundation of HTM theory [55]. George writes that HTM is "a memory system [that] exploits the hierarchical structure of [its] world." Specifically, the hierarchical structure of both space and time is explicitly

assumed in HTM. George continues, "[since] this method is a memory system that exploits hierarchy and time, we will call it by the name Hierarchical Temporal Memory (HTM)." Before describing how this memory system operates though, it is necessary to define the relevant terminology. This will be done in the context of a concrete example that George himself has used.

Preliminary Definitions and Considerations

To ground the definitions of HTM theory, George focuses on the classic problem in vision processing of invariant visual pattern recognition (IVPR). The biological analogy to this task is the mammalian visual system's ability to visually recognize objects in its field of view despite changes in the object's location, size, ambient lighting conditions, and other sources of noise. In machine learning, invariant visual pattern recognition is known as a classification problem. This is because IVPR tasks a computer with recognizing – or classifying – visual patterns regardless of nuance. This is a similar idea to what we encountered earlier in SA literature. Recall, Endsley and others told us [7][10]-[13], "[researchers] in many areas have found that expert decision makers will act first to classify and understand a situation, immediately proceeding to action selection." So the focus on IVPR is not unwarranted. Rather, we see in it a computational implementation of a processing task similar to one of the key ingredients of SA: classification. Of course, classification in IVPR is much more rudimentary than the classification aspect of a decision-maker's SA in a given context. We must therefore specify these levels of distinction as we proceed.

IVPR and other problems in machine learning naturally begin with the learning phase. As Mitchell reminds us [34], machine learning is "the study of computer algorithms that improve automatically through experience." So when speaking about machine learning, we are speaking about algorithms that do the learning. There are generally two types of learning algorithms: supervised and unsupervised. *Supervised*

learning algorithms use external – usually human-provided – information to refine the knowledge base they create from data. *Unsupervised learning* algorithms create a knowledge base directly from the data without external guidance. Some learning problems benefit more from one approach than another. For instance, unsupervised learning is more attractive when the 'correct' answer is not known *a priori*. Recall, we saw the need of an earlier computational SA implementation to "learn solutions to problems that are difficult for humans to codify with explicit rules or models." Brannon et al. went on to say [38], "In other words, [these solutions] can represent rules/decisions that are implicit in the training data." So when the training data is the only guide – the only description – of the phenomenon, unsupervised learning would be better-suited to find a solution than supervised learning. But in IVPR, we know what the 'correct' answer is when an algorithm is asked to categorize an object. So supervised learning can greatly aid the solution in IVPR. Still, it is not clear in biology which learning approach is employed in IVPR. George plausibly claims that biological creatures solve the IVPR problem generally by unsupervised learning. But many applications in machine learning have employed various kinds of supervised learning algorithms with remarkable success [91]-[97]. Of course, this is because supervision can greatly accelerate and improve the resulting classification abilities of the algorithms. Nevertheless, these are some minor distinctions between supervised and unsupervised learning.

To illustrate the difference even more, George treats the seemingly rudimentary task of a cat walking towards its milk bowl. In doing so, he also establishes some necessary terminology that will be useful as we proceed. This is an IVPR problem for the cat in recognizing the milk bowl as it changes its relative position to the bowl. But George treats this problem mathematically. Specifically, he assumes that each image on the cat's retina can be described with N pixels of data. So each image the cat observes can be thought of as an N-dimensional vector of data. These vectors can then be different points in an N-dimensional vector space (V). Consequently, the classification problem

done in IVPR is to group these vectors according to the identity of an object. For the cat, this means recognizing that each new image it perceives as it approaches the bowl belongs to a common class defined by milk bowls. There are many ways that each image of the milk bowl can differ from another, but the task of a learning algorithm is to find the common thread running through each of them.

But this image classification problem is not limited just to one object. Rather, this can be done for any number of objects. For example, Figure 17 shows two different binary objects. In this figure, each image of *Object A* differs from another one. The same is true of *Object B*. But the common thread running through each of these different images – or vectors – is what identifies one as *Object A* and the other as *Object B*. Without more specific terminology thus far, a 'thread' will temporarily refer to the perceived connection between images of a particular category. In supervised learning, these threads are formed both from the images themselves and user input. In unsupervised learning, these threads are formed just from the images themselves. Informally, supervised learning helps thread formation.

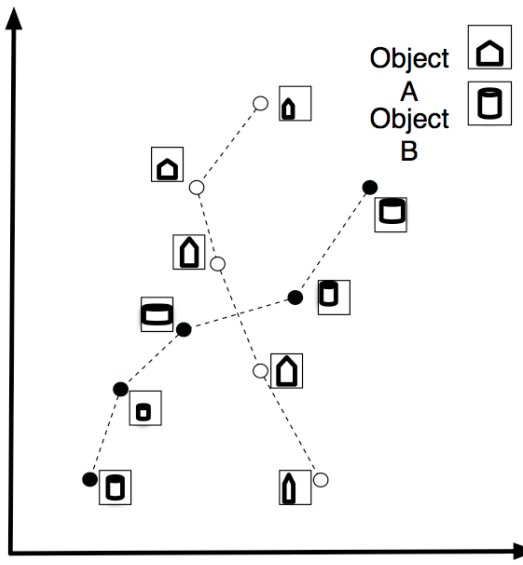


Figure 17: Graphical Representation of Object Categorization [55]

But unsupervised learning can also create these threads. The problem though is doing it correctly.

For either learning approach, George shows the difficulty in creating these threads with a simple demonstration. Figure 18 shows the possible threads that be created from a set of vectors. These vectors could represent images in the pictures context or general input vectors in some other context. The figure demonstrates only a few of a myriad of possibilities for threading the vectors.

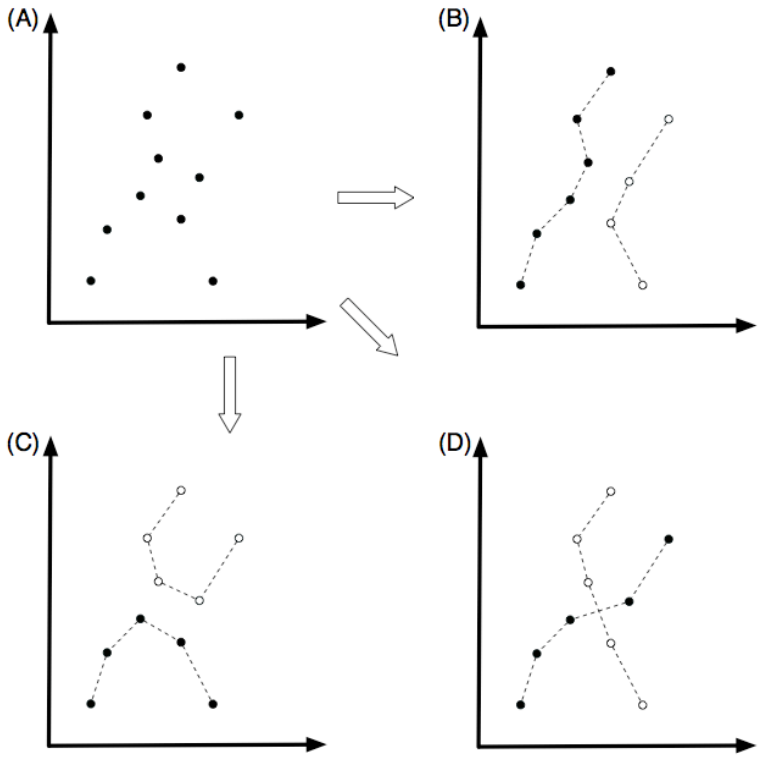


Figure 18: Possible Threads Through Different Vectors [55]

In fact, Figure 18 illustrates the well-known problem in combinatorial optimization of the traveling salesman [98]-[100]. Even in this simple example, the number of possible ways to thread these vectors is quite high. Furthermore, it becomes severely unmanageable to do this in the presence of more objects and more dimensions by which those images can vary from one another. So how does the cat do it? Does the cat solve a traveling salesman

problem when it learns what a milk bowl looks like? Or when it learns what other objects look like? This is likely not the case. So if not, then what mechanisms are at work when the cat learns how to thread different images of a milk bowl into an object category? Something must be guiding the process. So does this mean that supervision must provide the guidance?

Let us assume for a moment that supervision only guides the threading process. This means that each vector is connected to another one via a category label. For example, when proceeding along the points of the *Object A* thread, a supervisor would instruct the algorithm that the currently witnessed image belongs to the *Object A* thread. A similar process would happen for learning how to classify images of *Object B*. But George bluntly points out that "no one taught the cat about milk bowls by making it flip through pages of milk bowl pictures while shouting 'milk bowl' in its ears." Rather, George claims, the cat learns in an unsupervised manner that the different images on its retina as it approaches the milk bowl belong to the same category. No supervised instruction was needed to learn this. So if supervised learning need not direct the threading then other mechanisms must be at work.

The Role of Time in Learning

Time is one such mechanism that can guide the threading. George claims this but he is not unique in his attention to time as a guiding influence. For instance, slow feature analysis (SFA) is another technique that uses temporal slowness as an underlying principle in learning [101]. Informally, the idea of SFA is that images that occur close by in time likely correspond to the same object. George builds on this idea: "[when] there is relative motion between the cat and the milk bowl, different images of the milk bowl are produced on the cat's retina. However, these images occur close by in time. This means that different images of the same object are likely to occur close by in time compared to different images of different objects." In fact, such an observation is not surprising

because similar observations have contributed to the physical laws of causality. So by using temporal information, vectors can be threaded distinctly. Figure 19 shows this schematically with the same points used in Figure 18.

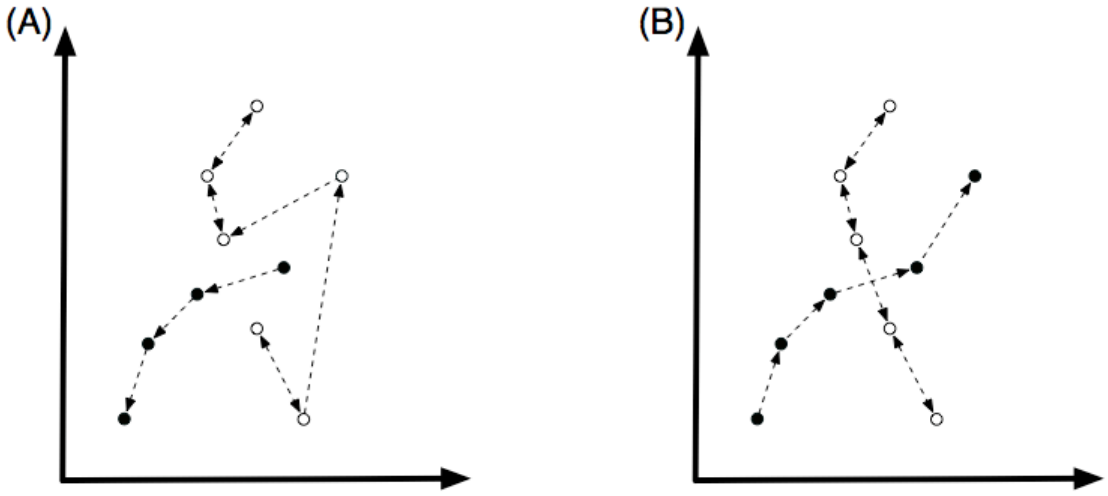


Figure 19: Illustration of the Use of Temporal Information in Threading [55]

In Figure 19, each sub-figure shows arrows between the vectors, indicating a possible sequence of how the images are observed in time.

Of course, this kind of learning opens the door to many possible false positives. Specifically, just because event *y* follows event *x* does not mean that *x* and *y* are causally related [102]-[105]. But we must recall the earlier literature survey in this regard. In particular, AFRL researchers in the first case of computational SA cited the fact that "Relational learning allows systems to exploit multiple tables in a database without the loss information that occurs in a join or an aggregation." They continue, "While the challenges are significant, so too is the potential payoff." So in no way should we discount the use of time for its danger of false positives. A consequence of the No Free Lunch Theorem of machine learning [89] is that every learning algorithm makes assumptions about the data from which it is learning. HTM is no different. This assumption necessitates though that training data for an HTM be temporally structured.

Recall, George writes, "if there is no temporal structure in the data, application of an HTM to that data need not give any generalization advantage." The assumption of temporal continuity between adjacent inputs during learning is the reason why.

The Role of Supervision in Learning

In light of temporal proximity sometimes being an inaccurate assumption during learning, there are other options available to accelerate learning. In particular, supervision can act as an accelerating force during learning. This is done by the user-provided supervision data providing the instructions for threading the vectors. In HTM, supervised learning works *together* with unsupervised learning, *rather than in place* of it. Some examples of this will be shown later. But the important conclusion to have thus far is that in HTM supervision can provide a correcting force to the false positives acquired during unsupervised learning.

But supervision is not always available as a learning option. In IVPR problems, supervision is possible because we know the name of the object *a priori*. For instance, *Object A* is a house and *Object B* is a cylinder, regardless of the shown distortions. But for instance in stability analysis of the Iraq context, this supervision data is not unique. Recall the UN Resolution from November 2006 in which the British UN representative saw signs of progress while the Russian UN representative did not. These assessments were the result of different kinds of situational awareness. At that time, the goals of each representative were not clearly the same and so their perception of the facts differed. Consequently, no unique supervision data can be used in this context to accomplish learning. Fortunately, there is no such liberty of interpretation in IVPR. All images the machine-learning user wishes to be categorized as *Object A* are specified and agreed upon *a priori*. From these images, it is hoped that the algorithm can learn to recognize other instances of this object's image. So we see from this distinction between the IVPR and the Iraq contexts that supervised learning overlaps strongly with goals. This issue

will be picked up later as we proceed to the shockwaves context. Right now, it is important only to note that supervision is not always an option in learning, though it can accelerate learning tremendously if it is.

More Definitions and a Concrete Example of Learning

Thus far, we have begun to get into George's HTM theory. Already though, we have seen how the specification of contexts and goals can impact the theory as it is extended to other domains of interest. To isolate issues one at a time, we will assume, following example from George, that the context is a screen of binary pixels and that the goal is to recognize the image on it. George calls this problem the "Pictures Problem." We wish to see from this example how hierarchical and temporal learning can be done in a simple context with a simple goal. In doing so, we will begin to see how an HTM can be an evidence-based inference mechanism. But a detailed account of that theory as told by George will have to follow, since learning is the fundamental step beforehand.

The Pictures Problem

The Pictures Problem is a simplified visual pattern recognition problem in which the goal is to use an HTM to learn invariant representations of a set of binary images, i.e., the context. Each image shown to the HTM is composed of a 32 x 32 grid of binary pixels. So each vector in this context has 32 x 32 = 1,024 components. During learning, the HTM must learn invariant representations of the same object. In other words, the task is to thread vectors into a particular category. These representations – or vectors – can be transformed, deformed or otherwise distorted from other members of the image category. These distortions are done to mimic the way that images can appear to be transformed due to relative translation, rotation, scaling and ambient noise. Consequently, the training dataset is a movie of images subjected to these variations.

The HTM instantiation used by George is a three-level network for the pictures problem. This network and the input image screen are shown in Figure 20. We will now describe some key terms shown in this representation of the pictures problem. The first one is as follows:

- The *receptive field* is the effective input area. Each input image appears in this area.

The entire context of the pictures problem is communicated through this input area. Consequently, the set of all input images integrated over time that appear in this receptive field forms the vector space, $V_{Pictures}$. Whether these images are used for training or later for inference, they always enter the HTM through the receptive field. Some other useful definitions are as follows:

- A *hierarchy* refers to the arrangement of nodes (indicated in Figure 20) in a network.

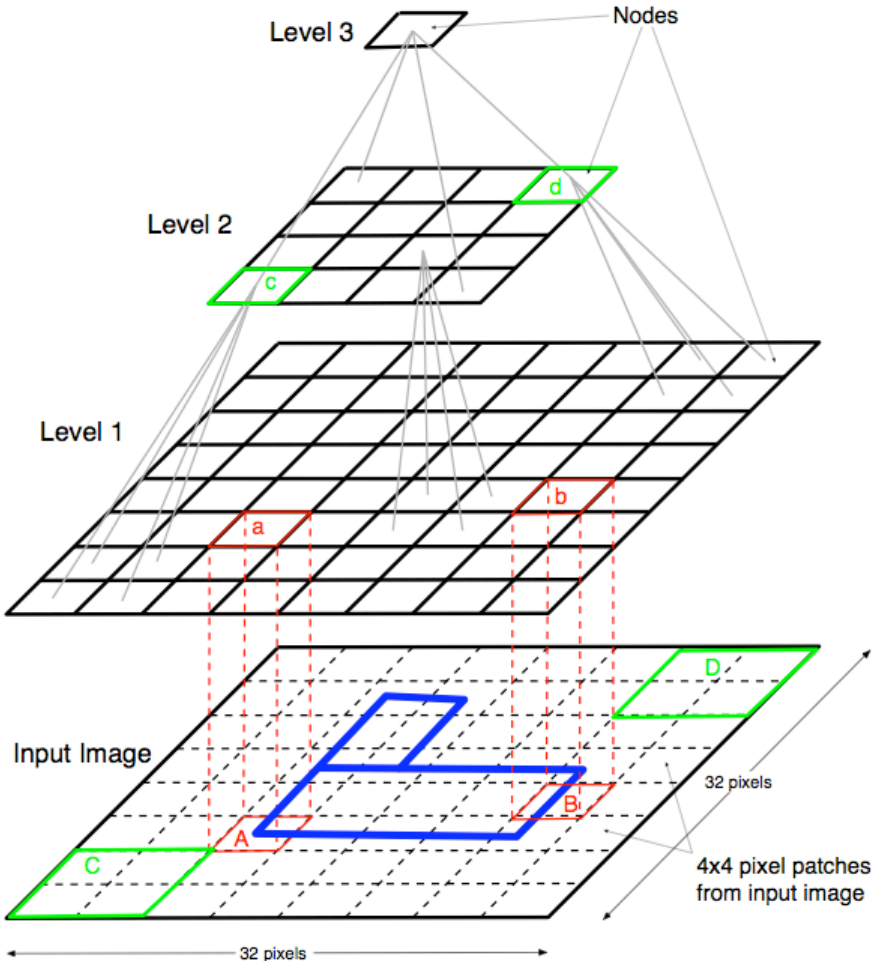


Figure 20: HTM Network Architecture for Pictures Problem [55]

- The *bottom-level* of the hierarchy refers to those nodes that are closest to the inputs of the receptive field.
- The *top-level* of the hierarchy refers to those nodes that are furthest from the inputs of the receptive field.
- The nodes connected immediately below a particular node are referred to as the *child nodes* of that node.
- The nodes connected immediately above a particular node are referred to as the *parent nodes* of that node.

Each node in an HTM executes the learning algorithms. Learning happens one level at a time. The learning algorithms of 'Level 1' learn directly from the receptive field. Once these nodes finish learning, their inference algorithms can produce outputs triggered by a replay of the bottom-up evidence in their receptive fields. This is the input then to the parent nodes, i.e., 'Level 2'. So the 'Level 2' nodes do not learn directly from the input image. Rather, they learn from the evidence-based inference of 'Level 1'. Informally, each node distills the experience of its environment into invariant representations so that higher-level nodes need not do so.

We can illustrate these terms and concepts in more detail with the network of Figure 20. We begin by noting that the "input image" constitutes the receptive field of the entire HTM network. Also, the receptive field of a subset of nodes can be specified. For example, the red squares marked 'A' and 'B' on the input image constitute the 4 x 4 pixel receptive field of the nodes marked 'a' and 'b' in 'Level 1'. Thus, each node's receptive field integrated over time would be only a proper subset of $V_{Pictures}$. We also see that the green squares marked 'C' and 'D' constitute the receptive fields of the nodes marked 'c' and 'd' in 'Level 2'. Therefore the size of the receptive field grows as we reach the top-level node, at which point all of the receptive field is in view. In Figure 20, the hierarchy refers to 'Level 1', 'Level 2' and 'Level 3', excluding the input image. The bottom-level nodes are the nodes in 'Level 1' because they are closest to the input image and the top-level node is 'Level 3' because it is furthest from the input image. The child nodes of the node marked 'c' on 'Level 2' are the four 'Level 1' nodes connected below to it. Similarly, this node is one of the children nodes of the only 'Level 3' node.

All of these nodes go through two phases operation: learning and inference. First, the bottom-level nodes are in the learning phase, during which time they receive inputs directly from their respective receptive fields; at this time, nodes above this level are turned off. Second, once learning has terminated, these bottom-level nodes are switched to inference mode. These nodes then classify the degree of membership of the inputs in

their receptive field to the memories they created during the learning phase. Then, these classifications of degree of membership are passed up to the parent nodes, serving as inputs to their learning and subsequent inference operations. The process repeats as the information is processed and passed up the hierarchy, along the way activating higher-level nodes, until each node is inferring the degree of membership of the inputs in its local receptive field. Since the same operations are repeated in each node, it is necessary to examine the details of the learning mode so that one can see how each node in a hierarchy condenses data. For now, we will only discuss the feed-forward inference, since it impacts learning as nodes in higher levels become activated. The feedback inference will be discussed once learning has been adequately described.

Although George describes the learning mode in terms of the bottom-level nodes receiving inputs directly from a region of the input image, the description of learning applies to all nodes of the hierarchy. For instance, instead of actual image fragments serving as the input, the inputs may be the feed-forward inference outputs of a child node. It happens though that it is easier to demonstrate the learning phase by observing the 'Level 1' nodes because human readers can visually identify the inputs from the image with ease, whereas this is not the case for inputs to higher-level nodes.

Since each image is binary in the Pictures Problem, any two vectors with a different component value constitute two distinct patterns. Mathematically, if $x_1, x_2 \in V_{Pictures}$ and $x_1 \neq x_2$, then x_1, x_2 are distinct patterns. Since the 'Level 1' receptive fields are only 4 x 4 patches of pixels, each 'Level 1' node is exposed to only a small pattern (i.e., a proper subset of the components of x) with whose neighboring patterns (i.e., other proper subsets of x) together make up the overall image (x). Each pattern (mathematically, a proper subset of x) to which such a 'Level 1' node is exposed is a 16-pixel vector. These components evolve in time over the course of the movie. Figure 21 shows what a 'Level 1' node in the pictures problem sees over a series of time steps when it is exposed to the

movie of images. Note that the 16-pixel vector is graphically viewed as a 4 x 4 binary patch, rather than a row of 1's and 0's for ease of interpretation to the human reader.

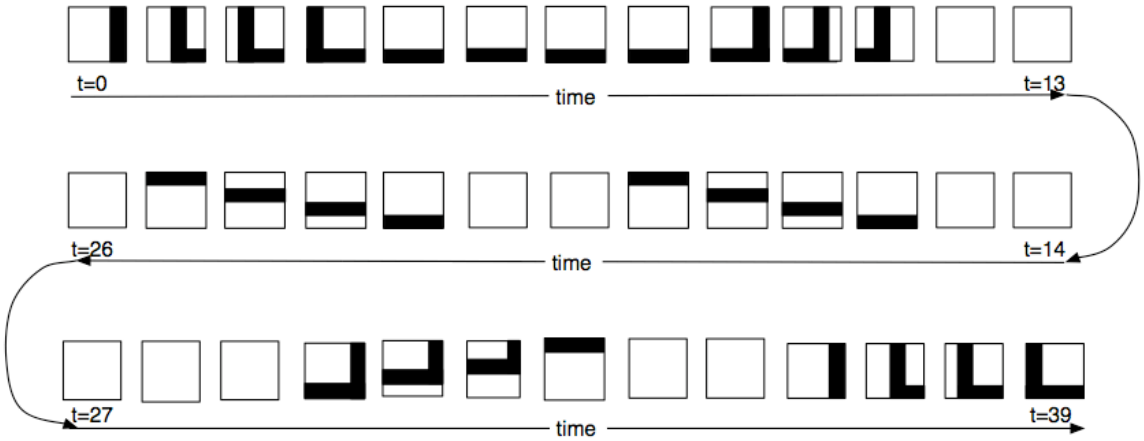


Figure 21: Time Evolution of the Receptive Field of a 'Level 1' Node – Pictures Problem [55]

As Figure 21 shows, the information with which the bottom-level nodes are provided is minimal when compared to the level of abstraction desired from the overall network. For now, let us focus on the 'Level 1' node. We will then be able to build up to the necessary level of abstraction, as George did. The task then remains for each node to learn something about the patterns it sees. In other words, the task is to create threads through vectors within a proper subset of $V_{Pictures}$.

Regardless of the level, there are three stages to the learning that occurs in each node and, for every input to the node (i.e., a subset of x), each of these operations occurs in sequence:

1. Pattern memorization
2. Transition probability learning
3. Temporal grouping

Each of these stages shall now be described in more detail, focusing on a 'Level 1' node as the illustration.

Each distinct pattern that is observed in the receptive field of the node is given a label, i.e., a number. Every distinct pattern is added to the memory of the node as a row of a matrix, *C*, in which each pattern is labeled *c1*, *c2*, etc. This label only indicates the row in which the pattern is stored. In general, the rows of *C* are certain subsets of *x* that satisfy a criterion by which their distinctness from one another is discerned. When the inputs are binary, the level of distinction is exact. Specifically, the pattern either exactly matches or does not match a pattern in stored memory. However, in general the inputs are not binary, and so a Euclidean distance calculation is then used to distinguish observed inputs from stored memory. This more general case will be discussed later, but for now let us continue with the assumption of binary inputs.

Once the distinct patterns have been identified (i.e., once *C* has formed), the probabilities of the transitions in time of one pattern to another are learned. A possible way to visualize these transitions and their associated probabilities is with a Markov graph. Figure 22 assumes that nine patterns have been learned in the pattern memorization stage; in the figure, each vertex of the Markov graph corresponds to one of these patterns.

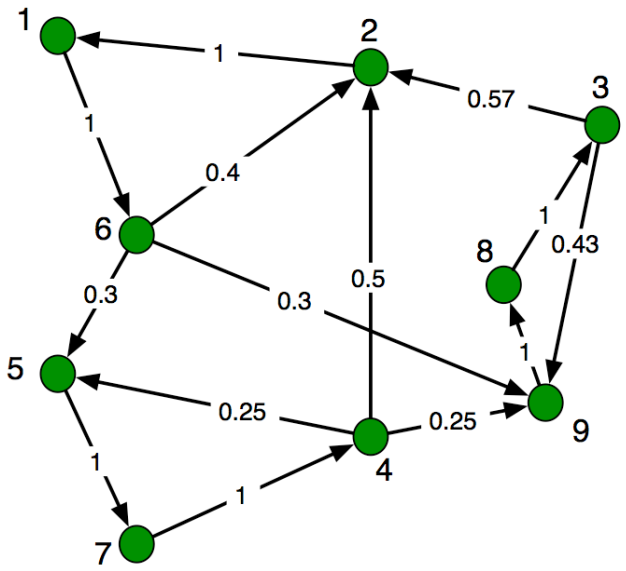


Figure 22: Markov Graph Showing Normalized Transition Probabilities Between Patterns [55]

Furthermore, the transition probabilities are written on the arrows connecting the vertices, where these probabilities are established based on how often pattern *i* preceded pattern *j*, where *i,j* are in the set made by the number of rows of *C*. The transition probabilities are normalized by dividing the number of transition events on the outgoing arrows by the total number of incoming transition events.

Once the Markov graph has been made, the final stage of learning is to partition the Markov graph into sets of vertices (i.e., patterns) that are likely to follow one another. Each of these partitions is a Markov chain of patterns and they are also called *temporal group*s. The method by which these vertices are clustered is agglomerative hierarchical clustering [106]-[108], and the clustering is adjustable by two parameters (*topNeighbors* and *transitionMemory*) when creating an HTM. To understand how temporal grouping works on inputs like those seen in Figure 21, George illustrates with part of the input used in the pictures problem. Figure 23 shows the time-evolving inputs fed into the node.

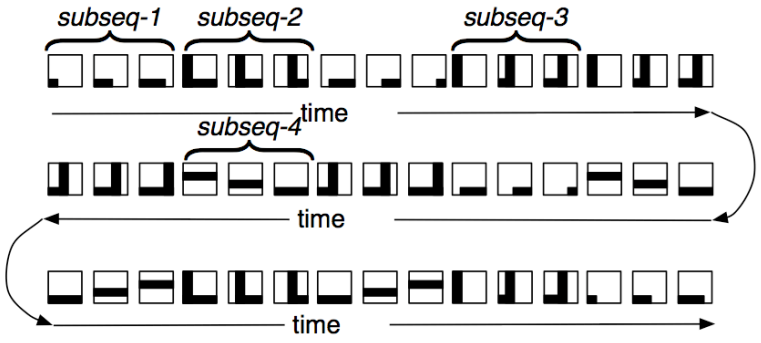


Figure 23: Input Sequence Presented to 'Level 1' Node – Pictures Problem [55]

The input sequence shown in Figure 23 is simplified from the pictures problem in that each of the subsequences (e.g., subseq-1, subseq-2, etc.) follows another according to a uniform distribution. Note that twelve of the twenty-four distinct patterns used are shown in Figure 24.

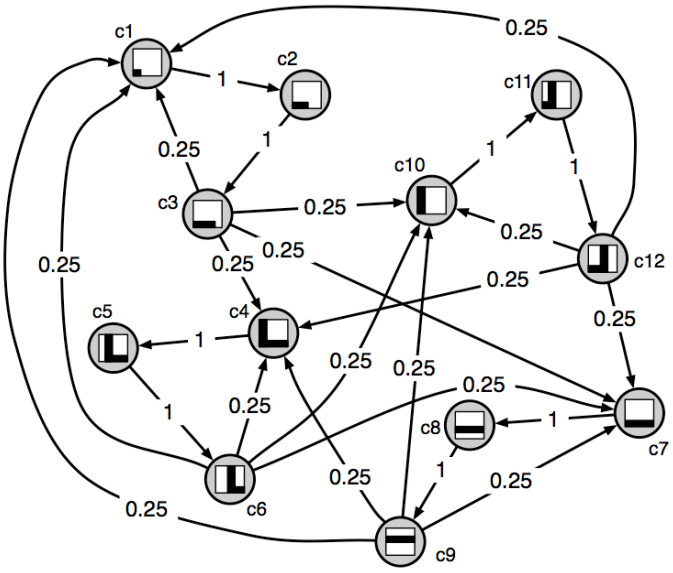


Figure 24: Markov Graph of Simplified Pictures Input

With the Markov graph in place, it is now possible to create *Markov chains* of patterns that are likely to follow each other, as shown in Figure 25. This process is also called *temporal grouping*.

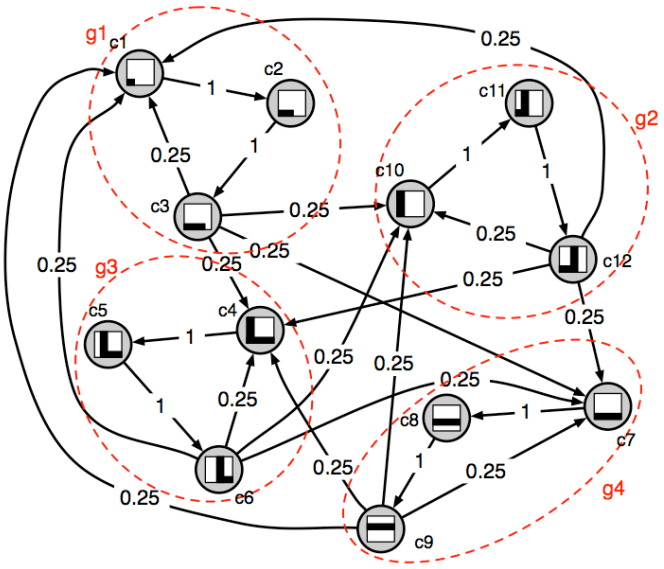


Figure 25: Temporal Grouping of Markov Graph from Simplified Pictures Input [55]

Each Markov-chain (*g1*, *g2*, etc.) has been determined by the two parameters mentioned earlier. For Figure 25, the parameters are set such that each Markov-chain is formed by considering one transition (*topNeighbors = 1*) and up to two time steps

(*transitionMemory = 2*) preceding the appearance of a given pattern. With this parameter choice, the node is able to determine that *g1* describes a horizontal line translating from left to right, *g3* describes an L-shape translating from left to right, etc. So we see from this demonstration how a node condenses data into a sequence of patterns. This is a fundamental aspect to the classification abilities of HTM.

There are two important points to realize though about the graphical representation of temporal grouping shown in Figure 25. First, note that the grouping shown in Figure 25 is not unique, since the parameters that determine grouping could have been more, or less, selective in its creation of Markov-chains. For instance, depending on the value of *transitionMemory*, the temporal grouping process could look back further in time or not far enough. Second, the geometry of the Markov graph has been modified so that groups happen to be easily indicated by circles on the graph. In general, the Markov-chains are not as easily visualized and as the number of learned patterns grows a Time Adjacency Matrix (TAM) is a more efficient means by which temporal groupings are visualized. In summary, as the preceding discussion alludes, temporal grouping completes the unsupervised learning within a specified subset of $V_{Pictures}$, whether it is proper or not.

In summary, the learning mode of each node is comprised of three stages that must occur in sequence. In short, the pattern memorization stage distills the raw input data to a subset of canonical patterns from which transition probabilities between these patterns are determined. It is then possible to temporally group patterns that are likely to follow one another in time. In other words, Markov-chains can then be created. With the learning mode suitably illustrated for now, it is possible to turn to the inference mode to see its role in learning. Specifically, we will consider the feed-forward inference.

In general, the inference mode uses the learned patterns and Markov-chains to do evidence-based inference on data presented to the node. The equations that execute these operations for the general case are shown in Table 3. Table 3 shows the equations for

feed-forward (rows 1 and 2) and feedback inference (rows 3 and 4). We will focus right now on those variables of interest to feed-forward inference. A full discussion can be found in George's work [54].

Table 3: HTM Inference Equations [54]

1) Calculate likelihood over coincidence patterns.	$y_t(i) = P(\bar{\ } e_t \mid c_i(t)) \propto \prod_{j=1}^{M} \lambda_t^{m_j}(r_i^{m_j})$ (2) where coincidence pattern c_i is the co-occurrence of $r_i^{m_1}$'th Markov chain from child 1, $r_i^{m_2}$'th Markov chain from child 2, $\cdots$, and $r_i^{m_M}$'th Markov chain from child M.
2) Calculate the feed-forward likelihood of Markov chains using dynamic programming	$\lambda_t(g_r) = P(\bar{\ } e_0^t \mid g_r(t)) \propto \sum_{c_i(t) \in C^k} \alpha_t(c_i, g_r)$ (3) $\alpha_t(c_i, g_r) = P(\bar{\ } e_t \mid c_i(t)) \sum_{c_j(t-1) \in C^k} P(c_i(t) \mid c_j(t-1), g_r) \alpha_{t-1}(c_j, g_r)$ (4) $\alpha_0(c_i, g_r) = P(\bar{\ } e_0 \mid c_i(t=0)) P(c_i(t=0) \mid g_r)$ (5)
3) Calculate the belief distribution over coincidence patterns	$Bel_t(c_i) \propto \sum_{g_r \in G^k} \beta_t(c_i, g_r)$ (6) $\beta_t(c_i, g_r) = P(\bar{\ } e_t \mid c_i(t)) \sum_{c_j(t-1) \in C^k} P(c_i(t) \mid c_j(t-1), g_r) \beta_{t-1}(c_j, g_r)$ (7) $\beta_0(c_i, g_r) = P(\bar{\ } e_0 \mid c_i(t=0)) P(c_i \mid g_r) \pi_0(g_r)$ (8)
4) Calculate the messages to be sent to child nodes.	$\pi^{m_1}(g_r) \propto \sum_i I(c_i) Bel(c_i)$ (9) where $I(c_i) = \begin{cases} 1, \text{ if } g_r^{m_1} \text{ is a component of } c_i \\ 0, \text{ otherwise} \end{cases}$ (10)

For the Pictures Problem though, George focuses on the feed-forward inference (rows 1 and 2 of Table 3) to describe learning. We will come back to top-down – or feedback– inference later.

In feed-forward inference, the node uses the learned patterns to quantify the degree of membership of each input. This can be seen in the equation for $y_t(i)$ in the first row of Table 3. In this equation, we see that $y_t(i)$ is the probability that the bottom-up evidence ($\bar{\ } e_t$) is in the node's receptive field, given the i^{th} pattern – also called *coincidence pattern* – in C. This degree of membership calculation eventually becomes an output of the node either to be read by the user or to be fed into a parent node as input. This feed-forward output is what the non-bottom-level nodes receive during learning. But the calculation of $y_t(i)$ is not the actual output. Rather, it is used to calculate the

distribution over learned Markov-chains. This happens in the second row of Table 3 via a dynamic programming variable, $\alpha_t(c_i,g_r)$. This dynamic programming variable provides a computationally more efficient way to calculate the $\lambda_t(g_r) = P(\breve{e}_0, \breve{e}_1, ..., \breve{e}_t \mid g_r)$, where $\breve{e}_0, \breve{e}_1, ..., \breve{e}_t$ is a sequence of evidence and g_r is a Markov-chain. Thus, the result of the feed-forward inference is a calculation of the probability that the sequence of evidence in the receptive field corresponds to the g_r Markov-chain. This quantity, $\lambda_t(g_r)$, is the feed-forward inference output of the node.

The utility of feed-forward inference during learning can be seen directly in the Pictures Problem. As described earlier, the learning and inference modes proceed in sequence from the bottom-level nodes to those of the top-level. Once they have completed learning, inference begins for the 'Level 1' nodes, at which time input to the 'Level 2' nodes is available for learning to begin in the second level. Then, inference begins for these nodes, providing input to their parent node ('Level 3'). Then the output from this node's inference stage is the classification of the object in the receptive field of the entire hierarchy. Although many of the results of the pictures problem are available in the thesis of George, there is a powerfully illustrative figure from this work that shows how the 'Level 1' nodes interact with their parent nodes. Specifically, it shows how feed-forward inference from the 'Level 1' nodes affects learning in the 'Level 2' nodes.

Figure 26 shows in detail how the learning and inference occurs for a simplified example from the pictures problem. This example demonstrates how the feed-forward inference equations (rows 1 and 2) of Table 3 are implemented.

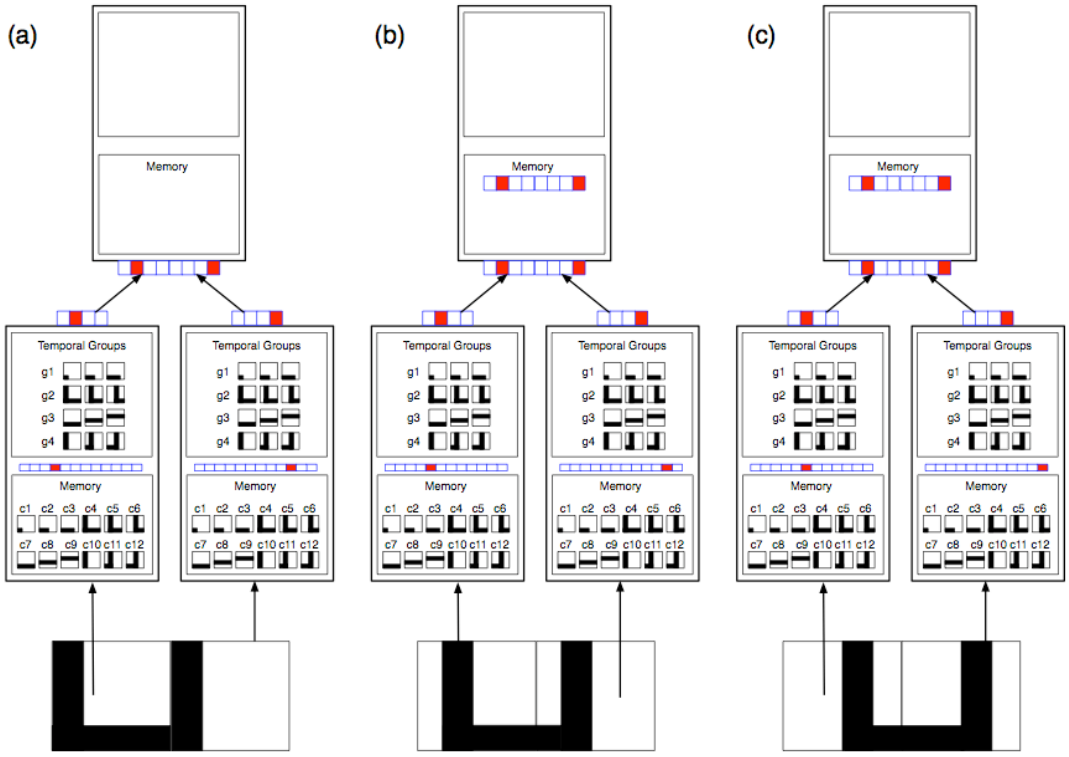


Figure 26: Information Flow During Learning and Inference Between Inputs, 'Level 1' and 'Level 2'

Figure 26 shows three sequential time steps (*a, b and c*) as a U-pattern translates to the right across two 'Level 1' nodes' receptive fields; then there is one 'Level 2' node above these two 'Level 1' nodes. In Figure 26, the 'Level 1' nodes have completed their learning and are now in the inference mode. But the 'Level 2' node is in its learning mode. In this case, both 'Level 1' nodes have the same memory of patterns (*c1, c2, ..., c12* $\in C_{1,1}, C_{1,2}$) and Markov-chains (*g1, g2, g3, g4* $\in G_{1,1}, G_{1,2}$). The 'Level 2' node is building its memory of patterns as time goes forward in parts *a, b* and *c* of Figure 26. In part a, the receptive field of the left 'Level 1' node shows an exact match to the *c4* pattern stored in its memory, while the receptive field of the right 'Level 1' node shows an exact match to the *c10* pattern. This is the calculation of $y_t(i)$ mentioned above for each of the 12 patterns. Now, since the *c4* pattern is in the *g2* Markov-chain, and the *c10* pattern is in the *g4* Markov-chain, the degree of membership of the left node's input to *g2* is 1.0 (i.e., 100%) and that of the right node's input to *g4* is also 1.0. Therefore, the output of the left

node in 'Level 1' is a vector with values [0, 1.0, 0, 0], while that of the right node in 'Level 1' is a vector with values [0, 0, 0, 1.0]. Once these outputs are concatenated, we have calculated the quantity $\lambda_t(g_r)$ mentioned above for each of the four Markov-chains. Of course, the use of binary inputs to 'Level 1' makes this calculation quite simple. This is because the evidence in each node's receptive field belongs entirely (i.e., 100%) to the group indicated by the component number in which a value of 1.0 appears. This concatenated output ($\lambda_t(g_r)$) provides the input now to the 'Level 2' node. Since this input has not yet been witnessed by the 'Level 2' node in part *a* of Figure 26, the input vector [0, 1.0, 0, 0, 0, 0, 0, 1.0] is added to the memory of stored patterns in the 'Level 2' node.

Now, we will see from this example how the feed-forward inference acts as a way to condense lower-level data during learning. Considering the time steps indicated by parts *b* and *c* of Figure 26, it is clear that the receptive field input has changed for both the left and right nodes in 'Level 1'. For example, in part *b*, the left node witnesses an exact match to the stored *c5* pattern and the right node witnesses an exact match to the stored *c11* pattern. This is another calculation of $y_t(i)$ for each of the 12 patterns in *C*. But despite a different set of inputs shown to the 'Level 1' nodes, the output from each node is identical to what it had been in part *a*. Why? This is because *c5* is a pattern of the *g2* group and *c11* is a pattern of the *g4* group. Therefore, the concatenated outputs from 'Level 1' are identical to the previously stored pattern of inputs to the 'Level 2' node. In other words, $\lambda_t(g_r)$ for each of the 4 Markov-chains does not change from time step *a* to time step *b*, despite $y_t(i)$ changing. The consequence is that there is no addition to the memory of patterns stored in 'Level 2' at time step *b*. Similarly, it can be seen that the same result occurs in part *c*. This is a fundamental performance feature of HTMs. Specifically, the combination of a child node's feed-forward inference with a parent node's learning allows the parent node to learn from temporally and spatially condensed data. This is an important illustration of schema formation within an HTM. Despite grossly changing inputs in the receptive field of a node (i.e., different image vectors x_1, x_2

and x_3 in $V_{Pictures}$ shown in parts *a*, *b* and *c*), the feed-forward inference of the 'Level 1' nodes has allowed a 'Level 2' node to recognize that these different inputs are connected to a common phenomenon. In this case, the common phenomenon is a U-shape moving across the receptive fields of the two 'Level 1' nodes. In other words, the 'Level 2' node recognizes that the different vectors are threaded by the 'U-shape' category.

But there are some nuances to learning that have been simplified thus far. First, the example described in Figure 26 has assumed that learned patterns exactly match inputs to the node. This is how the calculation of $y_t(i)$ was simplified. Second, the example has assumed that no pattern appears in two different Markov-chains. This is how the calculation of $\lambda_t(g_r)$ was simplified. In general, however, neither of these assumptions is true. So while this example from the Pictures Problem illustrates the interaction between learning and inference, there are some details to address.

George shows first what effects noisy inputs have on the learning process. For instance, if the inputs were noisy then the pattern memorization process described earlier would severely overload the memory as time increased. In other words, *C* would be too large to be of any use, i.e., it would not have condensed any data. But it is possible to pre-cluster the inputs using a k-means clustering algorithm [109]. This algorithm clusters inputs that are close to each other in terms of Euclidean distance. The criterion for 'closeness' of these inputs is tunable via a parameter (*maxDistance*) in each node. By using the k-means clustering algorithm, cluster centers reduce their movement in time as a given node reads new inputs. Therefore, pattern memorization is complete when these cluster centers have sufficiently stabilized. A graphical representation of the k-means clustering algorithm used in the presence of noisy inputs is shown in Figure 27.

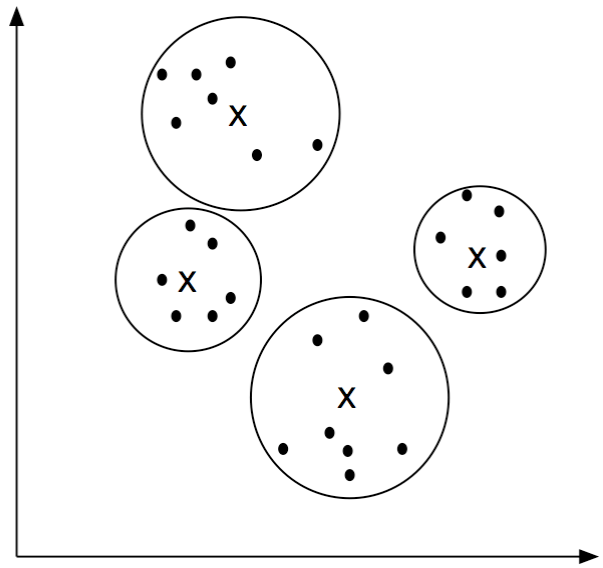


Figure 27: Illustration of K-means Clustering for Pattern Memorization During Learning [55]

Once they have sufficiently stabilized, the cluster centers are recorded in memory and given distinct pattern labels (*c1*, *c2*, ...). Consequently, this puts the number of clusters created for a given dataset in the control of the user. In other words, the condensation of data into a spatial schema is tunable depending on user needs. The rest of the learning process proceeds as before, with each vertex of the Markov graph now being a cluster center.

Second, George shows the effect on feed-forward inference when a pattern belongs to two or more Markov-chains. George writes, "Instead of signaling the membership of the input pattern in a group with complete certainty, the output is now a distribution that reflects the degree of membership of the noisy input pattern in each temporal group of the node" [55]. This is a more general description of the calculation of $\lambda_t(g_r)$. The consequence of this approach to feed-forward inference is that some degree of ambiguity is preserved in the output from the 'Level 1' nodes. This necessarily affects the learning of the parent nodes. For example, time steps *b* and *c* might not show the exact same $\lambda_t(g_r)$ coming from the 'Level 1' nodes if this is the case. But how does this affect the condensation process?

HTM has several embedded algorithms to facilitate the learning process in this regard. Specifically, some algorithms can be selected to filter the bottom-up data. This in turn helps the learning process of the parent node. Two such algorithms that employ this method are the *dot* and *product* algorithms. Before forming *C*, each of these algorithms employs a winner-take-all approach. Specifically, in the presence of a particular input, the Markov-chain with highest probability is set to 1.0, whereas all other groups' values are set to 0. George writes that this allows "the patterns seen during learning [to] match [either] the ideal patterns that are seen for the noise-less case" or those clusters generated by the k-means algorithm. George provides a lucent description of this process in Figure 28.

Figure 28 shows a node performing feed-forward inference when inputs are potentially noisy. Since the node represented in this figure has completed learning, it has a memory of patterns (*C*) and Markov-chains (*G*).

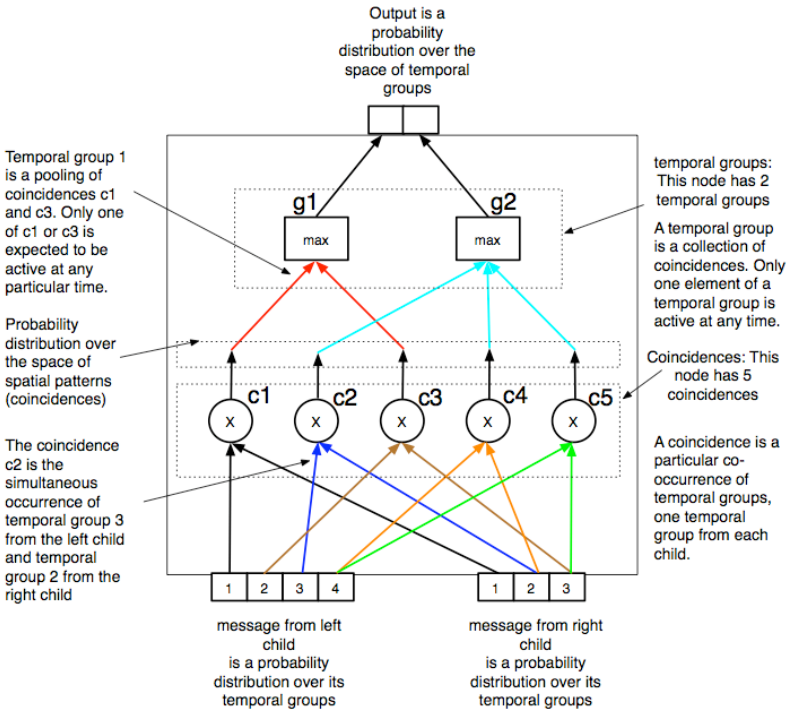


Figure 28: Inference Mechanism of Higher-Level Nodes [55]

Specifically, the node has learned five patterns (*c1, c2, ..., c5* $\in C$) and two Markov-chains (*g1, g2* $\in G$). The $\lambda_t(g_r)$ coming from the two child nodes tell us that the left and right child nodes have learned four temporal groups (the left node) and three temporal groups (the right node), respectively. Thus, a vector with seven components is the input (x) to the node represented in Figure 28. This node employed a winner-take-all approach when forming C. This is because each pattern represents the simultaneous activation of one group from the left node and one from the right. In other words, each pattern is the co-occurence – or co-incidence – of two Markov-chain activations. An activated group means that this group has the highest value of $\lambda_t(g_r)$ for the r^{th} Markov-chain. Consequently, by taking the maximum over the probability distribution inputted by its coincidences, each group can contain only one active coincidence at any time step. For example, let $\bar{e}_t = \lambda_t(g_r)$, where $\lambda_t(g_r)$ is from the child nodes. So when $P\left(\bar{e}_t \mid c_1\right) > P\left(\bar{e}_t \mid c_3\right)$, the *g1* Markov-chain is active. But this Markov-chain is also active when $P\left(\bar{e}_t \mid c_1\right) < P\left(\bar{e}_t \mid c_3\right)$. This alleviates then the potential problem of a pattern belonging to two or more Markov-chains. So we see from this example that, even in the presence of noisy inputs, feed-forward inference can still facilitate learning of higher-level nodes, as it did in the simple example from the Pictures Problem.

To summarize, we have seen now how HTM nodes can learn from time-evolving inputs. Both feed-forward inference in child nodes and learning in parent nodes combine as we proceed up the network. The end result of learning then is an HTM network that is capable of inference. This learning process is synonymous to what was called threading earlier. Now though, we know that 'threading' means finding Markov-chains from the input data. This time-evolving data mathematically exists in a vector space, $V_{Pictures}$, in which the HTM learns to thread certain vectors. In the preceding example, each thread has corresponded to an object, but in general this correspondence need not be true. Rather, if the context is not binary images and the goal is not IVPR then the Markov-

chains would correspond to some other condensation of spatial and temporal data. Of course, this is the main motivation behind studying these details right now. But it is not clear yet how HTM theory might aid a computational implementation of SA. Necessarily, the context matters and the goals matter. Furthermore, we have not yet seen how top-down inference works in HTM. Endsley and other human factors researchers told us of its importance in SA formation/maintenance, so we must examine to what extent it is done in HTM. However, having gone through the details of learning, we can now do this with relative ease.

The top-down inference in HTM facilitates a network's ability to classify novel inputs. The top-down inference equations are shown in rows 3 and 4 of Table 3. In these equations two crucial quantities are calculated, $Bel_t(c_i)$ and $\pi(g_r)$. $Bel_t(c_i)$ is the probability distribution over patterns in *C* due to the input coming from the parent above. $\pi(g_r)$ is the probability distribution over Markov-chains in *G* due to the same input from above. This $\pi(g_r)$ then serves as the top-down input for a child node to do the same calculations. In this way, ambiguities in feed-forward inference are resolved by the feedback. George illustrates top-down inference quite well with Figure 29.

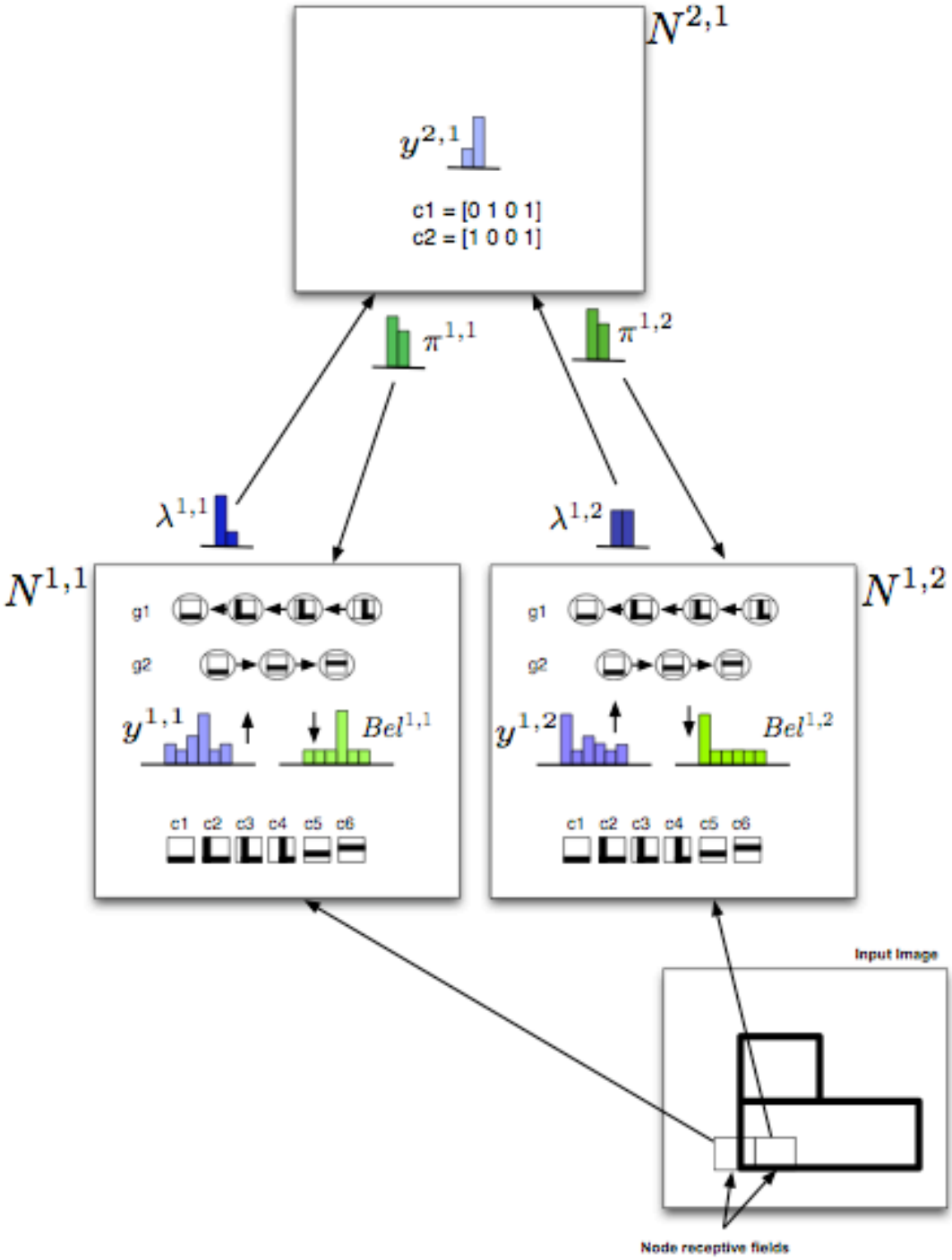


Figure 29: Top-down and Bottom-up Inference Example [55]

Here he shows how the feed-forward outputs, λ, are refined by parent nodes. For example, $\lambda^{1,2}$ indicates that both the *g1* and *g2* Markov-chains are equally probably, given the bottom-up input to $N^{1,2}$. This is because the horizontal line is the bottom-up

evidence ($\bar{e}_t$). This evidence though is also a coincidence pattern (c_1), which is in both *g1* and *g2* Markov-chains. So the probability distribution is split across these two categories. But the feedback inference from $N^{2,1}$ refines this distribution based on the $\lambda^{1,1}$ data because this output from $N^{1,1}$ indicates that the *g1* Markov-chain is more active than the *g2* one. Thus, the top-down input ($\pi^{1,2}$) from $N^{2,1}$ is a probability distribution that indicates a higher probability of *g1* being active. From this illustration, we see how both feed-forward (λ) and feedback (π) inference work in tandem during evidence-based inference in HTM.

To summarize, once learning has completed, an HTM network is an inference engine. It uses its condensation of data, i.e., its schema, to interpret novel data in light of what is has learned from the training data. If the novel data is somewhat related to the training data then conclusions about the novel data can be drawn. This process of interpretation happens by both bottom-up and top-down mechanisms of inference. The question now is to what extent these mechanisms are extendable to a computational implementation of SA. For instance, in the course of studying George's treatment of HTM theory, we have noticed the implicit agreement as to what data is relevant to a context's description. Furthermore, the goal was simply declarable in invariant visual pattern recognition because it was to recognize objects. But, in different contexts with different goals, it is not clear how HTM would be of use. We now shift our attention towards HTM's role in a computational implementation of SA.

Going From HTM to Situational Awareness

HTM is a detailed technical idea about cortical circuitry. SA on the other hand is a high-level description of processing mechanisms likely occurring in the brain when humans are engaged in complex and dynamic environments. In forming a computational approach to SA, we need to make progress by attacking the problems from both of these perspectives. There is no clearly linear way to go from HTM theory to SA. Rather, the

approach taken here requires both disciplines to meet in the middle as is shown in Figure 30.

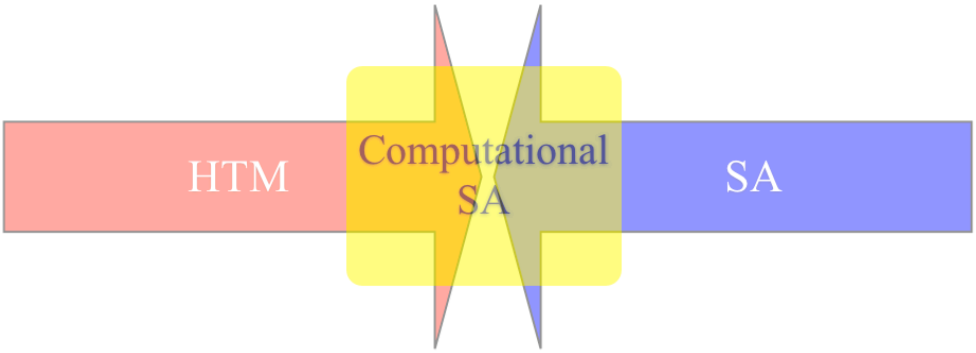


Figure 30: Illustration of HTM and SA Amalgamation

We have gone into much detail about HTM theory in the present chapter. Also, the literature review exposed us to the models of SA coming from human factors research. We must now see where the common ground is in computationally implementing SA. How much mapping is needed between HTM circuits and actual anatomic circuitry? Is there a mapping between HTM processing and those mechanisms seen in SA? These types of questions are crucial to our formation of a computational SA, and so they must be probed.

First, we should recall that SA is the result of a process, and so this must be considered when mapping it to HTM. Endsley reminds us [7], "It is first necessary to distinguish the term *situation awareness*, as a state of knowledge, from the processes used to achieve that state. These processes, which may vary widely among individuals and contexts, will be referred to as *situation assessment* or as the process of achieving, acquiring, or maintaining SA." Based on this statement and our knowledge of HTM thus far, situation assessment most aptly describes the learning mode of HTM operation. Specifically, during learning, the HTM is achieving and acquiring knowledge about the training data. Using this knowledge base then, the trained network recognizes novel data

with inference mechanisms that give probability distributions over likely Markov-chains. So during inference, one could think of an HTM's output as a reflection of the state of knowledge it has about the evidence it is observing. Specifically, assuming the top-level node's receptive field covers all data relevant to a given context (e.g., 'Level 3' in Figure 20), the output (λ) from this node would be the probability distribution over the most likely states of the context as described by the data below. In George's theory, these states are simply Markov-chains of patterns, and there is no specification in their definition that limits them from being perceived as such.

The actual act of situational awareness most relates to the inference mechanisms of HTM. This can be seen in more detail by recalling Endsley's model of SA from the literature review. Figure 31 zooms in on the three levels of SA that Endsley noted.

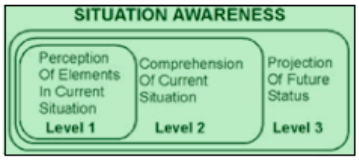


Figure 31: Central Focus of Endsley's SA Model [7]

Endsley reminds us that there are three phases to SA [7]: "Level 1 SA: Perception of the Elements in Environment, Level 2 SA: Comprehension of the Current Situation ... Level 3 SA: Projection of Future Status." Each of these steps is built on the previous one. HTM has little role in Level 1 because the user controls which relevant elements are input to the receptive field, i.e. perceived. Specifically, novel data that is within the scope of information covered by the training data is relevant to the current situation. Data that is extraneous to the situation is not considered. Our primary focus with using HTM in SA is on Levels 2 and 3. In particular, a trained network outputs its comprehension of the current situation (Level 2) via evidence-based inference. The mapping to Level 3 SA is less clear but there are certain aspects of it found in HTM. For instance, the feedback

inference (π) is somewhat of a predictive mechanism in that it refines the HTM's feed-forward inference output. Furthermore, there is flexibility in the type of inference algorithm used to calculate $\lambda_t(g_r)$ because some algorithms incorporate the temporal sequence of evidence up to a point rather than just the current evidence. An algorithm that does the latter would be the *maxProp* or *sumProp* algorithms, while one that incorporates a temporal sequence of evidence is *tbi* (time-based inference).

Of course, SA is a high-level account of processing mechanisms occurring in human brains, so we should expect some overlap between HTM and anatomical processing mechanisms. As discussed in the literature review, there are certain aspects of HTM theory that map to actual cortical anatomy. These mappings are recalled below in Table 4.

Table 4: Mapping of HTM to Anatomical Data [54]

#	Anatomical feature	Proposed computational role
1	Feed-forward thalamic projection to layer 4.	Storage and detection of coincidence patterns.
2	Layer 4 cell dendrites are mostly within layer 4. These cells make vertical projections to layers 2 and 3.	Bottom-up inputs required for the sequence likelihood calculation in equation.
3	Layer 3 cells with inter-columnar lateral projections to other layer 2/3 cells. Some of these cells send their outputs to higher order cortex.	Calculation of sequence likelihoods for feed-forward and feedback calculations.
4	Layer 5 cells with apical dendrites in the superficial layer 4 and bottom of layer 3.	Belief calculation without specific timing.
5	Layer 5 cells with apical dendrites in layer 1. These send outputs to subcortical regions and non-specific thalamus.	Belief calculation with specific timing.
6	Layer 6 neurons with apical dendrites in layer 5.	Computation of feedback messages for child regions.
7	Projections to layer 1 from higher level regions and from non-specific thalamic cells.	High level input is feedback information. Non-specific thalamic input is timing information for Markov chains.

For instance, the first row of Table 4 notes the mapping between HTM theory and thalamic projections to Layer 4. George writes, "Layer 4 is generally accepted as the primary feed-forward input layer to cortical regions." These cells implement the calculation of $y_t(i)$ as well as the learning of C. The fourth row of Table 4 notes the mapping of Layer 5 cells to the belief calculation over coincidences. These calculations correspond to the third row of Table 3. More of these mappings are discussed in George's work.

There are some shortcomings when HTM theory is mapped to anatomical data, but they do not detract from its use in computational SA. George acknowledges these challenges in both of his works on HTM [54][55]. He writes, for instance, that there are "[many] variations in cortical architecture [known] to exist [and that they are] not explicitly addressed in [his] model. Different cell types may be needed for short-term memory (not included in [the] model) and different types of attention. Inhibitory cells are needed to implement learning" [54]. But these details are by no means deal-breakers to HTM as a means to execute computational SA. Rather, the examples of SA seen in the human factors literature tell us that SA is a case-specific processing task. No two states of SA will be exactly alike. George builds on this point: "Given the behavioral flexibility and resilience of the cortex, we should expect some flexibility in the mapping between a theoretical model and its anatomical instantiation. If our model required a precise and unwavering mapping onto many unique cell types and their connections it is unlikely such a system could evolve." And, evolution due to data is precisely behind the formation of SA. So while HTM by no means explicitly models the human brain, there are many aspects of its information processing that can be useful for computational SA.

Furthermore, we might not even need a complete theory of brain function to reproduce many important results from implementing SA computationally. At this stage, we only seek an approach to computational SA that can serve as an aid, not as a replacement. In either case though, it need not be the exact way that a human does it. Rather, it might even be preferred *not* to be how a human does it, so that another perspective – perhaps a more quantitative one – be available when assessing the state of a given context. If we are to try HTM theory on the assessment – or categorization – of general states implicit in data, then we would have to rethink some aspects to HTM implementation.

To examine the mapping between HTM and SA in more detail, we return to Endsley's model of SA and note some differences. The first point worth noting is that SA

is an iterative process that occurs over time. Data is learned over the course of time and so a schema is formed. This schema is then used to assess a set of evidence relevant to a situation. HTM can do both of these steps with reasonable performance, as we saw with the Pictures Problem. But SA can be modified by a re-learning of new data in light of the failure of the schema. Right now, there is no automatic iteration yet by which HTM networks know that more learning is needed. Rather, the human user performs this analysis and initiates the relearning process if needed.

The second point worth noting is that the utility of HTM inference depends strongly on the perception of elements relevant to its knowledge base. If an HTM is tasked with inference on a data set completely outside the realm of its training, then it will output mediocre results, as does any machine learning algorithm. But Level 1 SA is specifically concerned with the perception of elements relevant to SA. So we must acknowledge here that we assume that the human user perceives relevant elements ahead of time. Specifically, there is a data pre-processing step that exists between the actual context and the HTM network's inference. This step gathers data relevant to the HTM network's knowledge base so that the output is of use. For example, the IVPR problem specifies the necessary elements worth perceiving, i.e., the visual data, ahead of time. However, if the size of the receptive field were to expand by one pixel in each direction, then the knowledge base created from a 32 x 32 pixel screen might not be of use anymore. Another example would be if a network trained from $V_{Pictures}$ vectors were tasked with inference of sound data. The sound data is not relevant to the network's knowledge base. So the user of an HTM must make sure that it is performing inference on data relevant to its condensed knowledge – specifically, its stored patterns and Markov-chains.

A third point worth noting is that SA is specific to a context and a set of goals. This point has arisen in the course of describing the Pictures Problem because both the goal and the context were implicit in it. The context was a 32 x 32 pixel binary screen –

also thought of as a 1,024-dimensional vector space – and the goal was to identify the objects. As Level 1 SA demonstrates, there is also some interaction between the goal and the data pulled from the context. But this issue was not considered in the Pictures Problem because it is assumed there that all binary pixels from the screen are of equal use in reaching the goal. For all contexts though in which HTM may be implemented, this need not be the case. For instance, in the Iraq context, only a finite proper subset of data is available to describe the context. So it remains to be seen how HTM can be of use when this is the case. Necessarily, in light of Level 1 SA being done by the human user, this puts a burden on the human user to find data that appropriately describes the context to be analyzed.

To take steps towards a computational SA, each of the three points described in preceding paragraphs will have to be considered. Along the way, human interaction will be needed, since none of these mechanisms are automatic. Though some schematics of data fusion processes bear resemblance to the information flow we will encounter [23], the most suitable description of our information flow comes from data mining [90]. Here, circles represent processes and rectangles represent results. This information flow has been embellished in light of our findings from SA and HTM theory, resulting in Figure 32. In this figure, we explicitly show what steps are done *a priori*, such as Level 1 SA, and what steps are capable of being done with HTM. While HTM is not a full predictive mechanism, there are aspects of it embedded in how it performs inference, and so that is indicated. Also, the roles of the human user are explicitly indicated. In particular, the human user prepares the data and evaluates the performance of HTM for the computational SA task at hand. If needed, the process can be repeated until satisfactory results are reached.

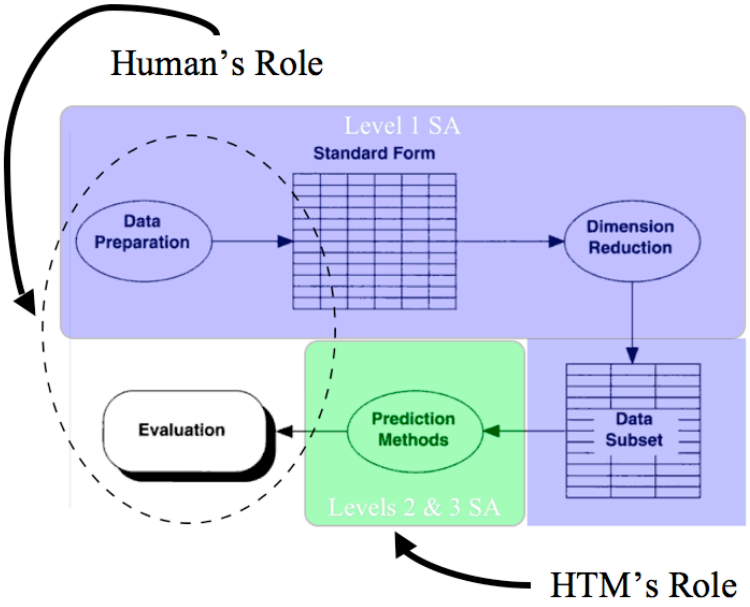


Figure 32: Initial Information Flow in HTM-based SA

Notice though that Figure 32 is an elaboration on the general picture we had in mind when laying down the research plan (Figure 16). After specifying some details on HTM and how it accomplishes certain aspects of SA computationally, we now have a more detailed process for going from data to SA.

As we move towards decision-making in the Iraq context, the utility of a quantitative approach for scenario categorization becomes increasingly clear. As Federico and Endsley remind us, classification of a scenario is the first and necessary step of decision-making. Furthermore, we can use the information flow of Figure 32 as a means to start this. So if we specify goals and relevant contextual data (Level 1 SA) then HTM might provide a means to classify the situation. Given the breadth of such situations, the hierarchical condensation of available data might provide a way to form a knowledge base of the scenario (Level 2 SA). Using this knowledge base, evidence could be presented to the network so that it can classify the situation according to its training experience. In other words, it can make predictions about this data (Level 3 SA). While

HTM is certainly not a suitable replacement for a human's information processing during such tasks, it could be of great use in this regard.

First Attempt at Situational Awareness with HTM

We would like then to extend HTM use to SA tasks and assess its performance. As noted in the research plan, we will do this via an evolutionary approach originating from a previous HTM application that was known to work. In doing so, we will take the necessary steps in creating a computational SA. Along the way, there will be some improvements to the information flow of Figure 32 as it is applied to computational SA in a given context. To see where these gaps lie, we will follow the information flow of Figure 32 in the course of this first implementation of HTM. By doing so, we will see what modifications to the method are needed.

The Waves Problem is a one-dimensional invariant visual pattern recognition (IVPR) problem, as opposed to the Pictures Problem, which was in two dimensions. It is a demo example that is available with the API on which HTM networks are built. The reason why we start from this HTM application is because it bears a keen resemblance to the Iraq context problem when looked at from a certain perspective. Specifically, another way to think of this problem is simply as an invariant pattern recognition (IPR) problem in which one time-evolving quantity is measured at N locations in space. By looking at the problem this way, we later ask if HTM can conversely perform IPR for N time-evolving quantities at one point in space. Seeing as our data on the Iraq context consists of N time-evolving metrics used to describe the stability conditions, this seems like a logical train of thought to follow. Only demonstration and testing will reveal whether it is conclusive.

The Waves Problem: Starting from One-Dimensional IVPR/IPR

In going through the Waves Problem, we will follow each of the steps shown in Figure 32. We first start at the data preparation step. This step is where we specify both our context and our goal, since both of these factors affect how the data is prepared. As mentioned, the Waves Problem is an IVPR problem that we are thinking of as an IPR problem. One property is measured at *N* points in one-dimensional space and the measurements of this property over time constitute our context. One could think of this context as temperature readings down a river, or longitudinal speed values down a lane of highway. However one thinks of it, our context (the river or the highway) is described by these values. Let us assume henceforth that these are temperature readings along a river. Our goal then is to recognize invariant temperature patterns as they propagate down the river. Recalling George's guidance on HTM generalization, this means that we have to train the network on data in which this occurs.

The data for the Waves Problem is created by a rudimentary Python code that simulates four Gaussian temperature profiles with a moving vertical offset. Some physical assumptions of the model are that heat does not diffuse, and that hot/cold points move down the stream unaltered. As time progresses, sensors later in the stream see what the earlier sensors had seen. In the context of this being a river, the factors that affect it are the Gaussian temperature profiles. The observations then that we have on the river amount to our observations of temperature at 32 locations along the river [110].

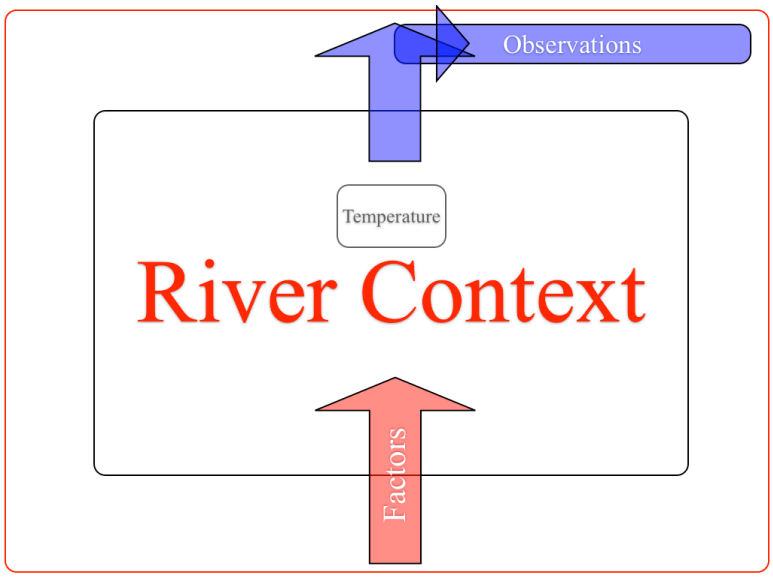


Figure 33: Dynamics of the River (Waves) Context

This is schematically shown in Figure 33. Notice that the factors are outside influences to the river context, of which we can only extract temperature data. The Python code creates the data such that each line of the data file is a time-slice of the temperature values across the $N = 32$ sensors. In each line, the temperature values are in space-separated format, so the data is in standard form. Recall, this is the second step of the information flow in Figure 32. Since all $N = 32$ dimensions of the data are needed, we bypass the dimension reduction step and can proceed directly to the prediction methods. So according to that figure, Level 1 SA for the River (Waves) context is now finished. Now, using this $N = 32$ component data, the HTM is tasked with classifying the four temperature profiles that proceed down the river, so we proceed to the implementation of Levels 2 & 3 SA.

It is important to note that the waves network only does Level 2 SA, not Level 3 SA. There is no prediction per se done with this network. Rather, for each piece of bottom-up evidence ($\bar{e}_t$) vector, the network performs inference. So this network does not perform Level 3 SA, i.e., there is no ability to make predictions.

Now, looking at its Level 2 SA, the original waves demo does this by using supervised learning. This is because the Waves Problem is solved with a supervised

network. Recall that for HTM, supervised learning is not done in place of unsupervised learning, but as an aid to it. This can be seen from the network topology shown in Figure 34.

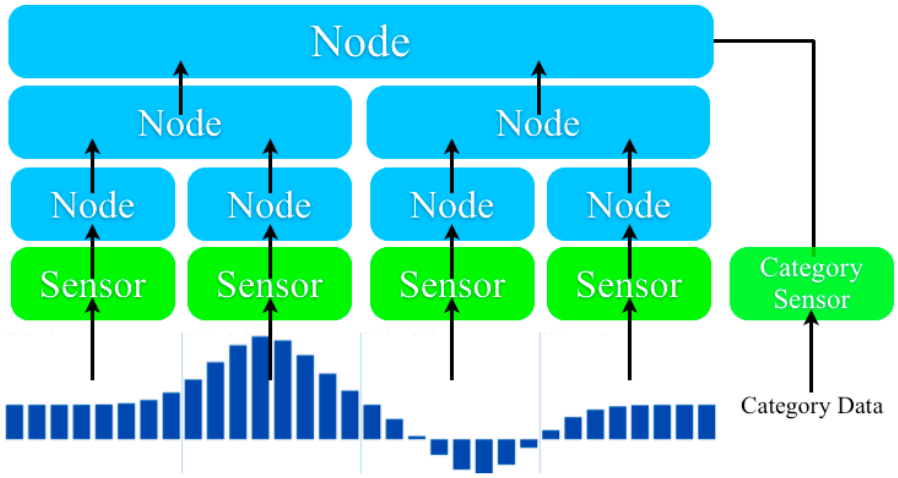


Figure 34: Schematic of Original Waves Network Topology

Here we see that unsupervised learning proceeds as described earlier above each sensor. Each of the four sensors has eight temperature monitors in its receptive field. There are four 'Level 1' nodes – one per sensor. And, there are two 'Level 2' nodes above these. Both 'Level 1' and 'Level 2' nodes perform unsupervised learning on the data in their local receptive fields, each creating their own *C* and *G*. But the one 'Level 3' node labeled *Top Node* actually receives category data as input too. This category data instructs the node as to which learned Markov-chains are to be matched with each of the four temperature profiles. Using this learning approach, the network represented in Figure 34 is capable of recognizing each of the four temperature profiles as they move across the network's receptive field. This is true whether or not there is noise in the data too, attesting to the robustness of the network. Both of these observations amount to the final step of the information flow, i.e., the evaluation. Thus, using a supervised learning approach, the goal of recognizing the four temperature profiles is attained with this

network. In other words, the Level 2 SA of this network is accurate with regards to both the River (Waves) context and any small random perturbations to it.

But for other contexts, supervision may not always be available. As a result, we want to see how much of an effect supervision has on the Level 2 SA of the network. This makes sense because we eventually want to apply unsupervised learning to discover implicit information about the Iraq context. So the first modification to a proven HTM is to remove the supervision from the network. To compensate for this lack of support, an additional level has been added as the bottom-level. The resulting network schematic can be seen in Figure 35. The complete network parameters are shown in an appendix (A.1).

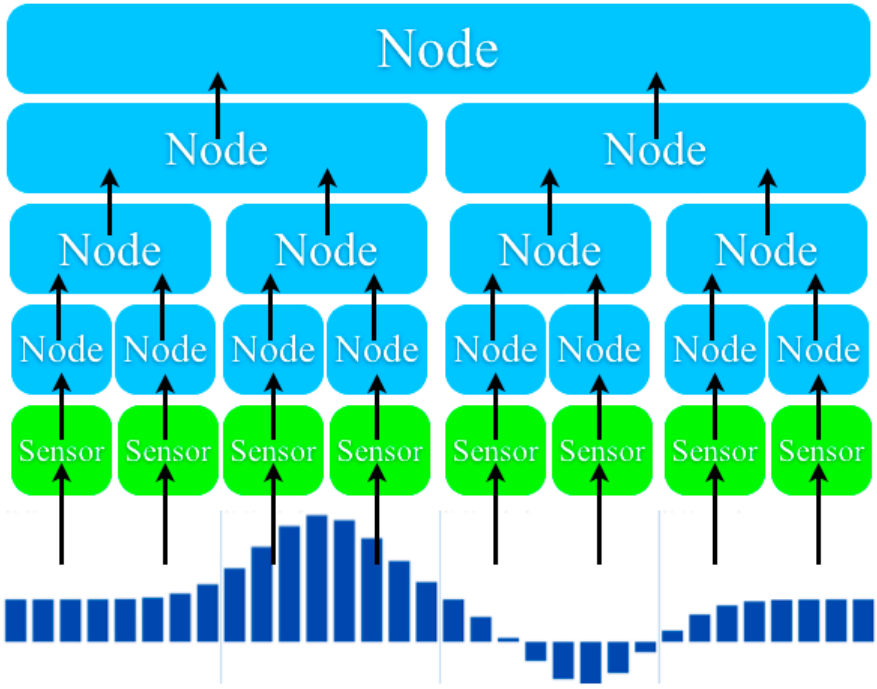


Figure 35: Schematic of Unsupervised Waves Network Topology

Now all nodes in the network are performing the same operations during learning. The feed-forward inference of the 'Level 4' node will be $\lambda_t(g_r)$, i.e., a probability distribution over $G^{4,1}$.

Looking at the highest probability over the top-level Markov-chains, we can see what the effects were on the recognition capabilities of the unsupervised network. This also amounts to the last step in the information flow of Figure 32 because we are evaluating the Level 2 SA, as we did before. Table 5 shows the resulting Markov-chain number (r_{max}) when we find $\max\left[\lambda_t(g_r)\right]$ at each value of $t \in [0,399]$.

Table 5: Highest Probability Markov-chains from Unsupervised Top-Level

Most Probable Markov-chain	Wave Type In Receptive Field	Time (t)
9	One-Peak	0-49 & 200-249
10		
11		
24		
0		
7		
3		
10	Two-Peak	50-99 & 250-299
11		
12		
6		
21		
25		
19		
26		
23		
24		
1		
5		
27	Peak-Trough	100-149 & 300-349
8		
28		
13		
14		
22		
15		
23		
29		
16		
0		
2		
0		
11	Peak-Trough-Peak	150-199 & 350-399
20		
4		
30		
17		
18		
5		

This is when the original training data serves as the bottom-up evidence ($\bar{e}_t$) in the receptive field. So we see from Table 5 that the network did not create Markov-chains that corresponded exactly to the four known temperature profiles. Rather, it created sequences of Markov-chains that repeated for each of the temperature profiles. For example, the one-peak temperature profile (or wave) was present in the receptive field $\forall t \in [0,49] \cup [200,249]$. So to some extent, the network's Level 2 SA recognized the different profiles of the original data. But how does it do with noise?

When noise is added to the data, there is minimal difference in the feed-forward inference of the unsupervised network. If we find $\max\left[\lambda_t(g_r)\right]$ again at each time point and make another table of r_{max}, then there is only one difference from Table 5. That occurs where *g11* is shown (during the peak-trough-peak temperature profile) in Table 5 because it was the second most probably Markov-chain at two time points. Instead, *g2* was the most probable Markov-chain. Consequently, we see from this test that the unsupervised network was nearly completely robust to random noise in the same way that the supervised network was. Consequently, the Level 2 SA is robust in this regard.

Notable Observations on the One-Dimensional IPR Problem

In following the information flow of Figure 32 for the preceding IPR problem, we noted several details that are not included. First, we saw that the first step of data preparation can be quite important. Of course, with the Waves Problem this issue was easily handled by the included Python code that generated the data. This code provided a rudimentary system dynamics model, if you will, on the river's thermal energy according to Figure 33. This model assumed a coarse graining by which temperature is measurable at $N = 32$ points along the river. The model used the aforementioned assumptions about heat exchange as the temperature profile passes down the river. However, in general, the modeling for a given context might be more difficult. Alternatively, for contexts on which data is prevalent, there might not be a need for such modeling. In general though,

it is worth noting that the data preparation into standard form is not as easy as it was here with the IPR problem.

Secondly, the model used to generate the data for the IPR problem did not need to be reduced in dimension. The original waves network was able to recognize each of the four temperature profiles without dimensional reduction (i.e., lowering *N*). Thus, while this step was bypassed in this simple IPR problem, in general, this is not the case. For instance, in the Iraq context, there is much data that violates the temporal continuity in time needed for HTM to generalize effectively. These dimensions of the Iraq context must be pruned before HTM can be used to attempt to create Level 2 (or even 3) SA.

Thirdly, we have rather awkwardly combined Level 2 and Level 3 SA into prediction methods in Figure 32. But we have clearly seen from Endsley that Level 2 SA is related to assessment of a current situation. Level 3 SA is prediction into the future. So we must separate these steps for the sake of having an information flow that is consistent across many implementations of computational SA.

One final point is that the evaluation step is likely followed with a repeat of the entire process. So we must indicate that the evaluation step can (and many times does) lead to a repetition of the entire process.

Modified Attempt to the Information Flow of HTM-based SA

In light of the preceding observations, we can modify the information flow for creating HTM-based SA. Figure 36 shows an information flow for HTM-based SA that accounts for these. Both the entrance point and the exit point are shown. At the exit, we have a computational SA. Adding to the convention from Figure 32, a triangle now indicates a question-based Boolean gate. Here, we see how the three levels of SA noted by Endsley map directly to processing steps in creating a computational SA. The figure uses terminology specific to HTM, whereas the previous one did not because it was merely a framework. Now though, we see explicitly where the user's role and the HTM's

role interact in the information flow. We will test this process as we proceed to other applications of HTM to SA of a given context. We see in this figure that the human's role is to prepare the data in such a way that HTM can be effectively used for generalization.

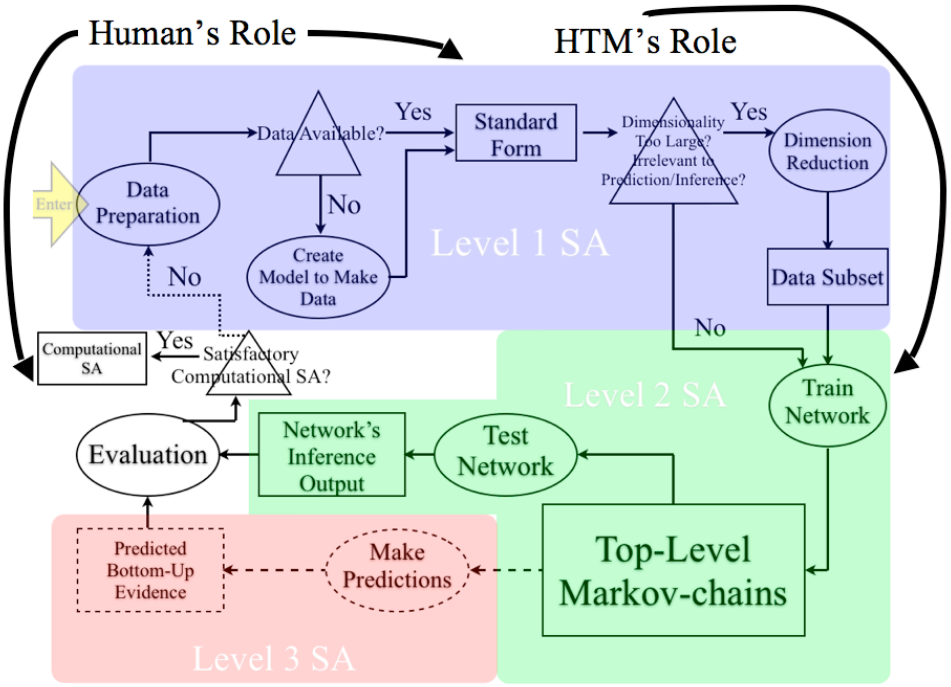


Figure 36: Modified Information Flow for HTM-based SA

Recalling from George earlier, this means ensuring that the data is temporally structured and is in a format suitable for the HTM learning algorithms. The question of dimension reduction did not arise in the river context. But it might be necessary to implement this step if the data has flaws in it that might inhibit training with an HTM network. So this step is included in Level 1 SA.

Proceeding then to Level 2 SA, we see in Figure 36 that the process of network training results in the top-level Markov-chains. These Markov-chains are a computational instantiation of a knowledge base necessary for SA. The process of network testing that results in inference output is the actual SA, since this process is stimulated with bottom-up evidence.

Level 3 SA is a more nebulous concept though right now. Some ideas will be posited later about how to accomplish it with HTM, but these ideas are not the main focus. Rather, Level 2 SA, i.e., the comprehension of pertinent elements, is the focus for us. Nevertheless, Level 3 SA and its specific result (predicted bottom-up evidence) are included above for completeness.

Both Level 2 SA and Level 3 SA need to be evaluated in light of predefined goals. This evaluation assesses the computational SA in light of these goals. If it is not satisfactory then the process repeats again. As the figure shows, the human role in data preparation is large right now. Consequently, the iteration of this flow cannot be automated with a computer. Nevertheless, due to the rapidity with which HTM algorithms train (on the order of 1-10s) and the evidence-based inference (nearly instantaneous), this process can be repeated with relative ease.

In what now follows, we shall implement this process on two problems of complex system analysis. The first will be in the shockwaves context and the second one will be in the Iraq context. The shockwaves context will be rudimentary in terms of its complexity, since it has been analyzed with simplifying physical assumptions. The Iraq context though will build on the first problem, and will highlight more assumptions that were implicit in our treatment of the shockwaves context. In light of these assumptions not being valid in the second context, a system dynamics approach will be used in conjunction with a comparative analysis to qualitative data.

CHAPTER 5

IMPLEMENTATION

In the previous chapter, we have refined our knowledge about HTM and have been able to reconcile this information with insights from the SA, data fusion and data mining literature. This culminated in an information flow process that can be followed as we attempt to analyze specific complex systems. For each analysis, both a context and a set of goals are first specified before entering the information flow.

As mentioned earlier, there are two problems that will be analyzed. Both extend from the work of the previous chapter and concern complex systems. Of course, the first one (the shockwaves context) is usually not thought of as such because of its approximate, yet highly accurate, treatment with simple physical laws. This analytical treatment will be used in two ways. First, it will provide a way to generate data that is usable for training and inference when real data is either absent or in short supply. Second, it will provide a foil to the analysis proposed here in which we develop a computational SA of the physical phenomena. This SA will be based off of simulated observations, as opposed to general physical laws.

The second problem (the Iraq context) will follow from the first. Although the connection does not seem obvious right now, the implementation of our method will show that there is some overlap. Specifically, we propose here a method by which such a complex system can be analyzed. Furthermore, we show that this framework can follow from how simpler physical systems are successfully analyzed. Only a continuation of the research plan can probe this claim in-depth. As we continue with steps 3, 4 and 5 in this

chapter, we will see to what extent this method for physical system analysis transfers to that of a complex system.

A Canonical Example: Shockwaves Context

The shockwaves context is the glue between the IPR contexts and that of Iraq. This is because we wish to demonstrate here that the rudimentary computational SA formed in the River (Waves) context can be extended to a shockwaves one. The reason this is a necessary step to take is because the data of the Waves context existed in a vector space representing the time-evolution of one property at $N = 32$ points in space. This was an example $V_{property}$ vector space. We wish to see now if a computational SA can be created for data in a $V_{properties}$ vector space. So instead of one property at N points, this vector space would represent N properties at one point in space. Can this switch be made? Does it yield a useful SA? This is the aim of exploring the shockwaves context. Ultimately, this experiment would pave part of the way to the Iraq context, since that is another problem represented in a $V_{properties}$ vector space.

In exploring the shockwaves context with an HTM-based SA, several issues will be explored. The first is the role of goals and what impact they can have on training, testing and evaluation. The second is the use of system dynamics modeling in generating training and testing data. Finally, we will see the effects that different preprocessing strategies have. All of these issues will be addressed in the course of executing the information flow created in the previous chapter.

Preliminaries to SA Formation: Goals and Context Specification

To follow the information flow of the previous chapter (Figure 7), we must first specify what is the context and what are our goals. The shockwaves context concerns the behavior of air in response to multiple passing supersonic bodies. Our goal is to recognize invariant patterns in its behavior. Recall that this is similar to what was done

with the River (Waves) context. But in that context, the patterns were learned and recognized according to temperature. Now however, our coarse graining is different. This is because the behavior of air – or any gas – has multiple dimensions of engineering importance. For example, the temperature is an important quality of the air to know in the presence of shocks [111]-[115]. Also, the density is important [114]. We could go on to find other properties of the air that are important in many engineering contexts. Our primary research goal with the shockwaves context though is to demonstrate that both $V_{Properties}$ and $V_{Property}$ are viable vector spaces for training and testing HTM-based SA. So we need only collect more than one property on the shockwaves context to demonstrate this. Even though we need to monitor only two properties to prove our point, we will see that a more accurate SA concerning the shockwaves context can be formed from monitoring four properties.

Of course, by saying 'more accurate', we mean in regards to a set of goals and so these must be specified. In fact, we shall probe the shockwaves context with two goals in mind. The first goal when forming SA will be to recognize different flow types. Is the air hotter? Is it more pressurized? Is it moving faster? These kinds of considerations all go into assessment of flow type because they are of general engineering importance [116][117][114]. The second goal when forming SA will be to recognize shocked flow. A shock occurs when the flow speed exceeds sonic conditions, causing a sharp discontinuity in many gas properties. These sharp changes have significant engineering importance because of their potential to do damage, cause instabilities and degrade efficiency [115]-[117]. Clearly, these two goals are prevalent when engineering within the context of moving air. So that is why we will focus on them as motivating goals to acquiring SA in this context.

Level 1 SA: Data Preparation, Modeling, Standard Form and Data Subset

As with the River (Waves) context, we do not have available data on the phenomena we wish to analyze. So as we enter Level 1 SA in the information flow (Figure 7), we realize that we must do what was done in that case: create a model to make the data. Once we have modeled the phenomena we then must put it into standard form so that it is amenable to HTM learning. Finally, we will assess whether the dimensionality is too large or whether the included data is relevant. The end result of this process is a data subset in standard form that is ready for HTM learning. Knowing that we must generate the data from a model, we proceed now to describe this first step in forming Level 1 SA.

Modeling

In modeling the shockwaves context, we are in effect creating it by making its data. This means that a set of assumptions, which we shall now describe, is necessary. First though, it is important to note the altered significance of the dimensionality of the data. This data exists in a vector space as it did in the Waves context. But in the transition from SA of a $V_{property}$ vector space to that of a $V_{properties}$, we are changing the meaning of the vector space's dimensionality. In the Waves context, physical space is assumed to be spread over $N = 32$ points, thus making a curve (or river). But in a $V_{properties}$ vector space, we assume that physical space is collapsed down to one point at which N properties describe it. The level of coarse graining then determines which and how many properties describe this point. Though N is still the dimensionality of a vector, its real world significance has changed.

In the shockwaves context, if we assume that N properties describe one point then we must make assumptions about how they change in time. The assumption of local thermodynamic equilibrium (LTE) posits that all four properties change instantaneously with one another [114][118]. For instance, LTE assumes that the non-equilibrium

relaxation of excited vibratory modes in diatomics (e.g., N_2, O_2 in air) is negligible. The removal of this and other time-dependent phenomena is what culminates in the assumption of LTE. This is a useful approximation to make in many cases of engineering importance, and so we will use it here for our purposes. A consequence of this assumption is that we also assume a close enough succession of time points such that the gas' internal energy does not go into chemical dissociation. For instance, this ensures that chemical dissociation also does not occur after a shock. Consequently, it can be assumed throughout the model that air is made of 79% diatomic nitrogen (N_2) and 21% diatomic oxygen (O_2). Finally, since we are concerned only with the time-evolution of one point, a one-dimensional flow analysis will suffice. In other words, the flow is assumed to move in only one direction.

With these assumptions in place, it is possible to set up some external factors to have an influence on this defined context. In other words, we want to simulate dynamic situations. Recalling the Waves context, this was done with simulated waves moving along a river. From a modeling perspective, this was simple to do because it only required a sequential horizontal displacement of a Gaussian in time. A Python code was able to generate this data easily. Now, however, our factors are different because we are examining air's behavior in the presence of passing supersonic bodies. Despite this difference, the representation of the shockwaves context looks quite similar to that of the Waves one (see Figure 37).

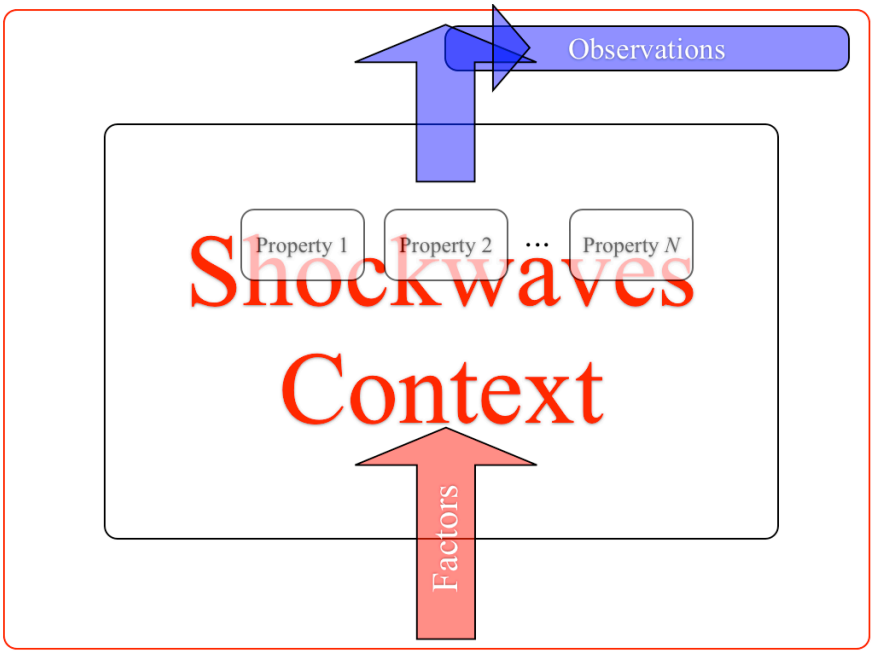


Figure 37: Dynamics of the Shockwaves Context

Here we see that instead of temperature being the only observable, there are N observables. These observable properties define the coarse graining and are the means by which SA is ultimately formed. We assume in modeling the shockwaves context that the only external factors affecting it are supersonic bodies. So we must account for their effects when modeling this context. This requires us to focus on the analysis offered by the one-dimensional normal shock equations [114][118] every time a supersonic body is present.

The one-dimensional normal shock equations are well understood and will provide us a means to generate data specific to our context. To generate the desired dataset, four gas properties are assumed to be observable initially: pressure (P), density (ρ), enthalpy (h), and flow speed (u). Only these four upstream properties plus an assumed equation of state are necessary to know the downstream properties after the shock. When there is no shock, the properties are assumed to remain unchanged. For either case, the mass, momentum, energy, and state equations are shown in Equation 1.

Note that the state equations are consistent with the assumptions we have made thus far (LTE and purely diatomic air).

Equation 1: Mass. Momentum, Energy and State Equations

$$\rho_1 u_1 = \rho_2 u_2$$
$$P_1 + \rho_1 u_1^{\,2} = P_2 + \rho_2 u_2^{\,2}$$
$$h_1 + \frac{1}{2} u_1^{\,2} = h_2 + \frac{1}{2} u_2^{\,2}$$
$$P = \rho RT$$
$$h = \frac{7}{2} RT$$

With Equation 1, it is possible to propagate an initial condition forward in time. To simulate the dynamics of passing supersonic bodies, it is assumed at every tenth time point that the flow goes supersonic by a random fraction of the local sound speed. So every tenth time step, external factors shown in Figure 37 affect the shockwaves context. It is important to note that any interval could have been chosen, as long as there was one time point after the shock available for equilibrium recovery. Physically speaking, the amount of time for each time step could be such that ten time steps occupy less time than that which is needed for kinetic energy from the shock to go into or to leak out of either the chemical or vibrational modes of the gas. We have chosen ten time steps for ease of analysis. The variation of flow speed versus time is now shown in Figure 38.

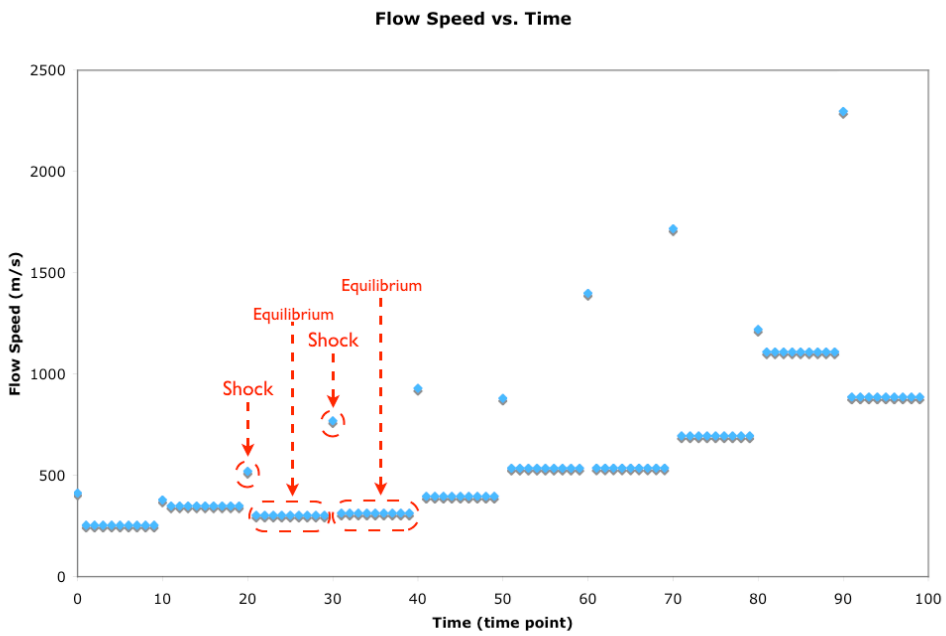


Figure 38: Variation of Flow Speed with Time Over Simulated Flow Conditions

Here we indicate two of the ten simulated shocks with their equilibrium recovery periods. With flow conditions illustrated by Figure 38 and an assumed initial condition of standard atmosphere at 2000m below sea level (also, arbitrarily chosen), Equation 1 uniquely dictates the time-evolution of the initial conditions. For instance, the properties' variations over the first 21 time steps are shown in Table 6 and the variations for the subsequent time steps can of course be calculated from Equation 1. The remainder of the data is available in an appendix (B.1).

Table 6: First 21 Time Steps of Equilibrium Normally Shocked Flow Properties

time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
0	1.28E+05	1.48	2.74E+05	411.91
1	2.24E+05	2.40	3.27E+05	253.19
2	2.24E+05	2.40	3.27E+05	253.19
3	2.24E+05	2.40	3.27E+05	253.19
4	2.24E+05	2.40	3.27E+05	253.19
5	2.24E+05	2.40	3.27E+05	253.19
6	2.24E+05	2.40	3.27E+05	253.19
7	2.24E+05	2.40	3.27E+05	253.19
8	2.24E+05	2.40	3.27E+05	253.19
9	2.24E+05	2.40	3.27E+05	253.19
10	2.24E+05	2.40	3.27E+05	377.88
11	2.51E+05	2.61	3.37E+05	348.48
12	2.51E+05	2.61	3.37E+05	348.48
13	2.51E+05	2.61	3.37E+05	348.48
14	2.51E+05	2.61	3.37E+05	348.48
15	2.51E+05	2.61	3.37E+05	348.48
16	2.51E+05	2.61	3.37E+05	348.48
17	2.51E+05	2.61	3.37E+05	348.48
18	2.51E+05	2.61	3.37E+05	348.48
19	2.51E+05	2.61	3.37E+05	348.48
20	2.51E+05	2.61	3.37E+05	519.51

With the time-evolving gas properties providing the raw data of an example of $V_{properties}$, it is now possible to see if an HTM network can learn different flow types according to our predefined goals.

But before proceeding to that analysis, we should also use our modeling framework to create testing data. This will be of use in assessing computational Level 2 SA with an HTM because it will aid us in assessing the generalizing capabilities of the SA. This data will be made from perturbations of various sorts to the training data. For instance, we will perturb the data by constant values and random percentages. These perturbations can all be considered additional factors that affect the shockwaves context. In using them for testing, we will be testing the robustness of our SA under these conditions. Some example perturbation datasets are shown in an appendix as well (B.2 and B.3).

Standard Form and Dimension Reduction

Having modeled the context, we must coalesce the data into a standard form that is suitable for analysis with HTM. The standard form for HTM analysis is either a comma-separated or space-separated data file. Considering that the model was created in Excel with a macro, it is rather easy to put the data into standard form. The first line of the data file must denote the number of properties. Then the time-evolution of the properties is simply cut-and-pasted into the file below it. For all implementations done here, the space-separated format has been used. Some examples are shown in an appendix (B.4 and B.5).

Also, another consideration will be whether or not to transform the data before HTM analysis. In the Pictures and Waves problems, there was no need to do this because the data values were all on a comparable relative scale. But as we see from Table 6, that is not the case now. Considerations about if and how to transform the data will necessarily impact the learning and inference. For instance, should we normalize the data by a maximum? Or, should we transform the data with a logarithm? Issues with these transformations will be case-specific, and we will see how these choices play out as we proceed.

With the data in a standard form, we then ask whether the dimensionality is too large. When we consider this aspect to the data, it is important to consider not only what implicit information is to be extracted, but also by what means the algorithms do this. In the shockwaves context, we will show how these issues interact. For instance, what will be the effects on SA if only two properties are considered instead of four? These issues will also be addressed as we proceed. After considering the dimensionality issue, we will have a data subset suitable for training. Consequently, this ends Level 1 SA and we proceed now to Level 2 SA.

Level 2 SA: Training and Testing

With a model and a process in place to make data, we now turn to forming Level 2 SA from this data. This SA can then be tested on perturbed data, giving a sense of its generalizing capabilities. We will test the formed SA by looking at many examples within the shockwaves context. First, since our main goal in this experiment is to analyze SA formation from a $V_{properties}$, we will consider an $N = 2$ vector space. This vector space will be formed of two-dimensional vectors whose values of temperature (T) and flow speed (u) evolve in time. The SA will be tested then on perturbations to these vectors. We will also look at the effects on SA formation from transforming the data by a logarithm. Second, we will also see what effects there are on SA if $N = 4$ properties are used. Perturbations of this data will also allow us to test the SA. This data will also be transformed by a logarithm to allow us to assess the effects on SA. These two versions of the shockwaves context experiment will allow us then to evaluate the formed Level 2 SA.

Experiment #1: Level 2 SA with Two-dimensional Data

The first experiment concerns forming Level 2 SA from two-dimensional data on the shockwaves context. Notice that this is somewhat of a test on the effects of dimension reduction from Level 1 because all four properties were needed to solve Equation 1. Here though, we are seeing if it is possible to meet out goals with just two properties. This necessitates recalling our goals in this analysis. First, we wish to recognize different flow patterns, as defined by the properties' values. Our second goal is to identify shock patterns in the flow. But which properties are most important of the four that were used to solve Equation 1 as we proceeded in time? Due to the second goal, uniform flow speed (u) is an important property to include. But for the second goal, this property is only important in relation to sound speed. Since the chemistry is frozen, we can assume that sound speed is nearly directly proportional to temperature (T) only. For equilibrium flow, sound speed is in fact dependent on two state variables, so our use of only one (T) is

necessarily not the complete story. But we shall see what consequences this choice has on the resulting SA. So *u* and *T* are the properties forming the $N = 2$ $V_{properties}$. These two properties are also acceptable in light of the first goal. For instance, we mentioned earlier the importance of temperature in engineering applications; however, flow speed is equally important [111]-[118].

Having prepared the data in standard form, we can train and test an HTM. The first datasets to be used in Level 2 SA formation consist of the *T* and *u* values in their SI units. An example of these datasets can be found in an appendix (B.6). Since we take an evolutionary approach to designing our HTM, the HTM that served in the River (Waves) context is our starting point. Since the vectors in our experiment have only $N = 2$ components, as opposed to $N = 32$, only a two-level network is needed. Its parameter settings are shown in an appendix (B.7), but this network is very similar to the network used in the unsupervised waves context. The major difference is that the third and fourth levels of that network have been removed. Also, the values of *maxDistance* and *sigma* have been adjusted to account for the magnitudes of *T* and *u*. The receptive field coverage of the resulting nodes has then been adjusted to give us the Figure 39 schematic.

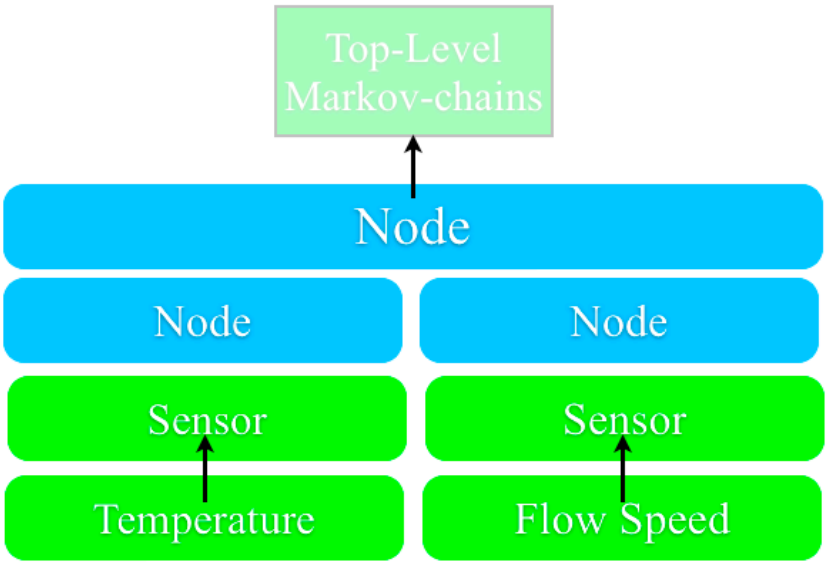
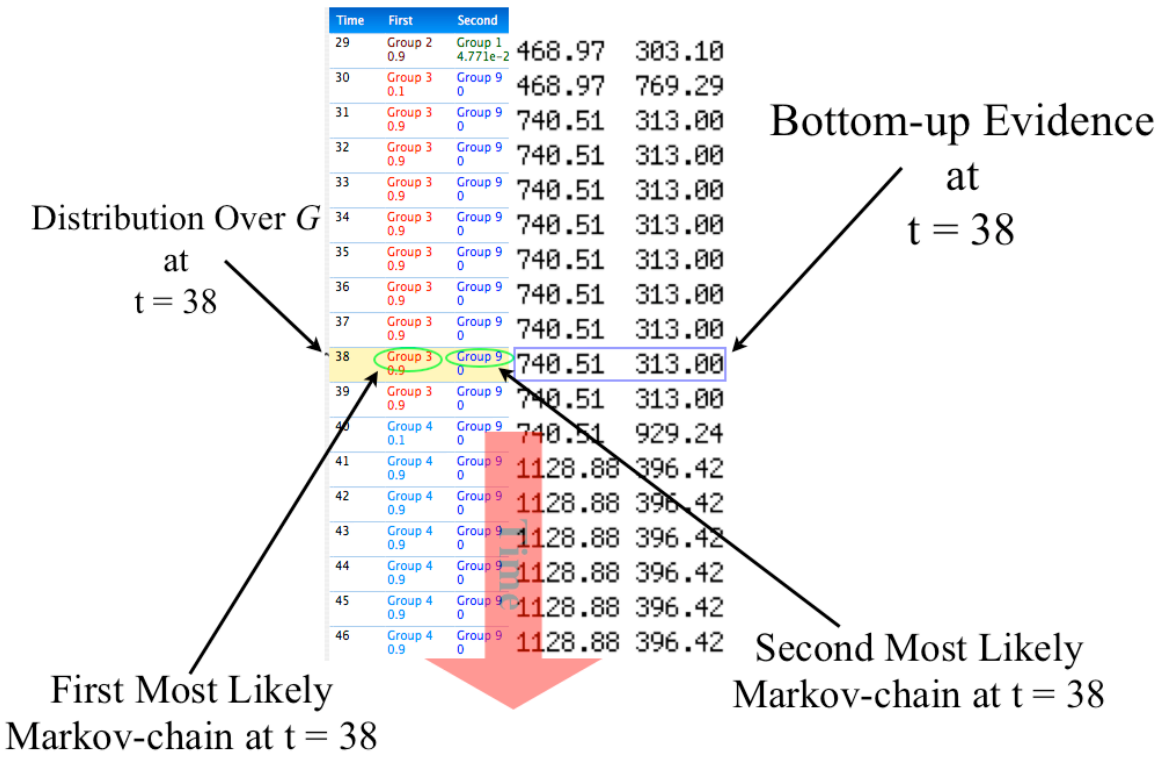


Figure 39: Schematic of Two-Dimensional Unsupervised Shockwaves Network

By comparing Figure 39 to that previous network schematic (Figure 35), the major topological adjustments can be seen. Comparing them in the appendices (A.1 and B.7), the specific changes in parameter settings can also be seen.

After training the network, the next step in our information flow is to look at its output (Figure 7). The top-level node learned twenty patterns from which it learned ten Markov-chains. This is the schema into which it has condensed data on the $N = 2$ $V_{properties}$. Having trained, the network is now able to do evidence-based inference. The top-level node's output from this inference task then is a probability distribution over these ten Markov-chains.

So we look first at the top-level node's inference ability on the training data. What flow patterns does it recognize? Does it recognize shocks? These questions amount to the next step in the information flow (Figure 7) in which we examine the network's inference output. As with the River (Waves) context, the crucial quantities are each Markov-chain and its likelihood in light of bottom-up evidence. Below (Figure 40) we can see the bottom-up evidence and the top-level's value of $\max\left[\lambda_t(g_r)\right]$ for each value of $t \in [29,46]$.



Time	First	Second		
29	Group 2 0.9	Group 1 4.771e-2	468.97	303.10
30	Group 3 0.1	Group 9 0	468.97	769.29
31	Group 3 0.9	Group 9 0	740.51	313.00
32	Group 3 0.9	Group 9 0	740.51	313.00
33	Group 3 0.9	Group 9 0	740.51	313.00
34	Group 3 0.9	Group 9 0	740.51	313.00
35	Group 3 0.9	Group 9 0	740.51	313.00
36	Group 3 0.9	Group 9 0	740.51	313.00
37	Group 3 0.9	Group 9 0	740.51	313.00
38	Group 3 0.9	Group 9 0	740.51	313.00
39	Group 3 0.9	Group 9 0	740.51	313.00
40	Group 4 0.1	Group 9 0	740.51	929.24
41	Group 4 0.9	Group 9 0	1128.88	396.42
42	Group 4 0.9	Group 9 0	1128.88	396.42
43	Group 4 0.9	Group 9 0	1128.88	396.42
44	Group 4 0.9	Group 9 0	1128.88	396.42
45	Group 4 0.9	Group 9 0	1128.88	396.42
46	Group 4 0.9	Group 9 0	1128.88	396.42

Figure 40: Inference Output of Top-Level as Function of Bottom-up Evidence into Network

We see here that the value of $\max\left[\lambda_t(g_r)\right]$ changes in response to the shock. For instance, at $t = 30$, *g3* is the most active Markov-chain, whereas at $t = 29$ it had been *g2*. This trend repeats for $t \in [39,40]$, $t \in [49,50]$, etc., indicating that the network certainly reacts to the shock as its bottom-up evidence ($\bar{e}_t$). However, for $t \in [30,39]$, *g3* is still the most active Markov-chain. This trend repeats also for $t \in [40,49]$, $t \in [50,59]$, etc., indicating that the network also reacts to the *subsequent* equilibrium flow in the same way. Of course, air properties preceding and following a shock are not the same, so why would the network recognize these two types of flows as the same? First of all, each time the flow speed goes supersonic ($t \in \{0,10,20, \ldots 90\}$), $\max\left[\lambda_t(g_r)\right]$ is only *0.1*, as opposed to *0.9*. This indicates the low confidence the network has in the shock conditions being related to the equilibrium conditions to follow. Of course, ambiguity is built into this inference task because only one out of two available properties has changed. We know

from our model that at the shock, only one out of four total properties changes. But the network did not learn this way and it will not infer this way either. All it learned was that there were ten unique changes in the properties at $t \in \{0,10,20, \ldots 90\}$ that created ten unique patterns during learning. The equilibrium recoveries were just as dramatic of a change in both properties ($t \in \{1\text{-}9,11\text{-}19,21\text{-}29, \ldots 91\text{-}99\}$) and they created ten other occurrences. Together with the previous ten, this made the twenty coincidence patterns the top-level node learned. But there is no other information with which the network can connect the shock conditions to the equilibrium recoveries. Consequently, the HTM-based Level 2 SA does not seem too helpful for either goals. It somewhat recognizes shocks in the flow, but not uniquely. And its recognition of different flow types is at odds with what we know *a priori* about how different a shocked flow is from one that is in post-shock equilibrium. If performance on the training set were satisfactory then we would proceed to look at inference on perturbed data. But since this is not the case we will not do so.

Rather, let us now see how the SA changes if supervision is given during training on the same data. This supervision data indicates at each time point whether the flow is shocked or not. So our goal here is to understand the difference between shocked and un-shocked flow from the data. Recalling the discussion on supervision from the previous chapter, we should expect this to help the learning and consequent inference of the network, given this goal. The exact parameter settings for this network can be found in an appendix (B.8). But the main change to the previous network (Figure 39) is that the top-level node can receive category data to map to its Markov-chains. We then train the network on the same $N = 2\ V_{properties}$ with this additional category data and assess results.

With this increased guidance atop the unsupervised learning algorithms, the top-level node mapped its Markov-chains to the two categories provided by supervision data. In inference on the training data, as expected, the network recognized shock and non-shock conditions perfectly (B.9). This is an expected result, but it does not tell us much

about the Level 2 SA formed by this network. For instance, how well does it categorize data not used in training?

We can test this Level 2 SA with perturbations to the original training data. So we perturb the dataset by a random value between 5% and 10% of the original value. We then recalculate the category data to provide a check during inference. It is important to note that the perturbation category data assumes both γ and R have not changed, making the perturbed a purely a function of the perturbed T. For equilibrium flows, this is an approximation because we have assumed frozen chemistry. An example of this perturbed data is shown below for $t \in [0,13]$ (Table 7) and the rest of it is given in an appendix (B.6).

Table 7: Sample of 5-10% Perturbed $N = 2$ Bottom-up Evidence

time	**temp**	**u**
time step	K	m/s
0	277.69	377.72
1	384.91	236.13
2	377.64	240.40
3	331.53	231.45
4	336.90	239.80
5	379.50	237.59
6	379.49	233.21
7	383.64	232.73
8	333.65	235.61
9	384.44	273.76
10	326.39	413.60
11	349.64	366.16
12	395.17	370.44
13	407.00	321.68

Proceeding to inference of this data, we observe that the network recognizes the flows flawlessly until $t \in [60,99]$, during which time the network is completely unable to recognize the flow. We can look here at the transition point to note this change in performance. Specifically, looking at the bottom-up evidence for $t \in [50,68]$ (Figure 41), we can see that the perturbations grow larger as the values of T and u increase. As time increases, these perturbations become too large to be considered similar enough to the

patterns stored in the first level of the network ($C^{1,1}$ and $C^{1,2}$). Consequently, the top-level node is unable to categorize the feed-forward inference from the bottom-level nodes, as we see in Figure 41.

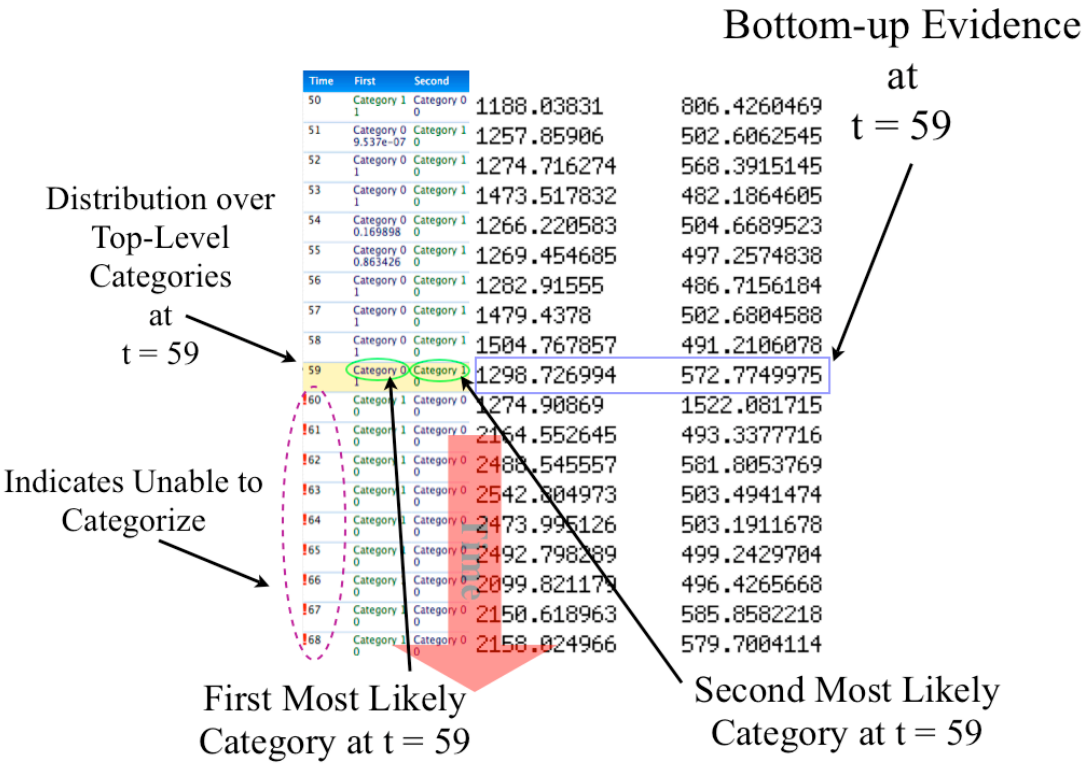


Figure 41: Degrading Categorization as Perturbations to Bottom-up Evidence Get Too Large[1]

The trend seen for $t \in [60,68]$ then continues through $t = 99$. We can see from lower-level nodes why this happens. For example, the node above the temperature sensor outputs a null probability distribution over its Markov-chains ($G^{1,1}$) as early as $t = 22$. After $t = 27$, all of the non-zero feed-forward input to the top-level node comes from the node above the u sensor ($N^{1,2}$). This can be seen below in Figure 42.

[1] Note that significant figures have not been truncated to experimental accuracy for the purposes of reproducing the results obtained here. This convention is used throughout the work.

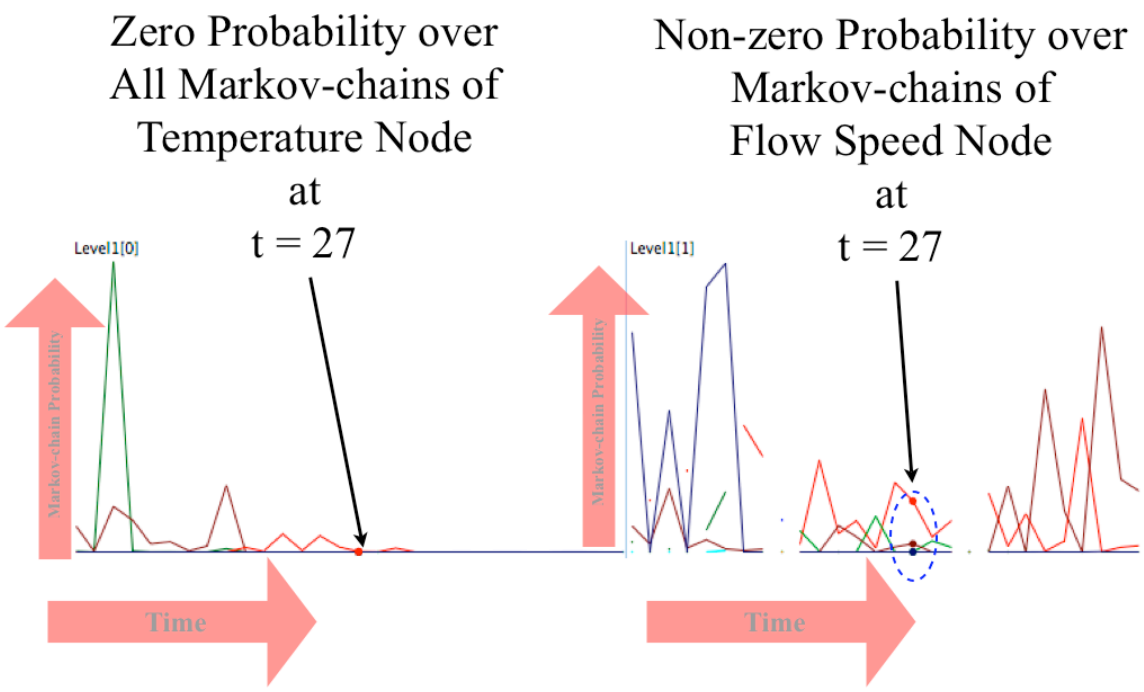


Figure 42: Degrading Recognition in Bottom-Level Node Above Temperature Sensor Starts at *t = 27*

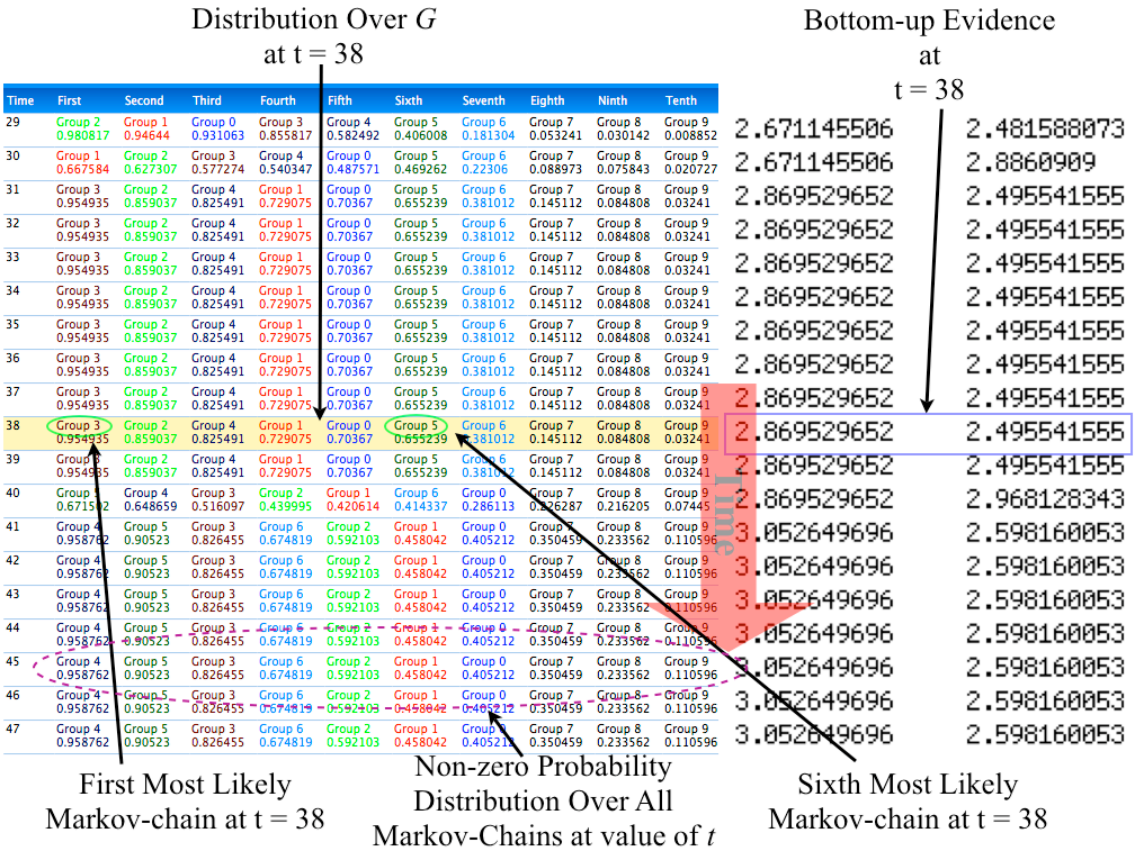
So while the network displays commendable performance in recognizing shocked and un-shocked flow at each time point, there is a falloff in performance as perturbations become larger in magnitude. It is for this reason that we proceed to attempt a transformation of the $V_{properties}$ by a logarithm.

We proceed to a log transformation of this data to assess learning and inference effects. We choose this transformation because it somewhat linearizes the actual changes in the values. For instance, infinity normalization would not do this enough in light of the changes in both temperature (T) and pressure (P). So we use the same two properties as data now but subject them to a $\log_{10}$ transformation (see an excerpt of this data in Figure 43). This transformation can be thought of as an aspect to Level 1 SA, in which we prepare the data for training to occur in Level 2 SA. This particular transformation is done because it decreases the numerical difference between post-shock temperatures/speeds from pre-shock values. Additionally, it has the same effect when

perturbations of the data by 5-10% are done. Consequently, we should expect better inference performance during $t \in [60,99]$.

We begin first with an unsupervised network. This is nearly the same network used for the untransformed data, except for an adjusted value of *maxDistance* in bottom-level nodes to account for the transformation. The complete parameter settings can be seen in an appendix (B.10). As with its predecessor, the top-level node of this network learned twenty coincidence patterns ($C^{2,1}$) and ten Markov-chains ($G^{2,1}$). With training complete, it is now possible to assess inference performance.

During inference of the training data, we see some differences from the un-transformed case. First, the distribution over Markov-chains in $N^{2,1}$ is more spread out (see Figure 43).



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth	Tenth		
29	Group 2 0.980817	Group 1 0.94644	Group 0 0.931063	Group 3 0.855817	Group 4 0.582492	Group 5 0.406008	Group 6 0.181304	Group 7 0.053241	Group 8 0.030142	Group 9 0.008852	2.671145506	2.481588073
30	Group 1 0.667584	Group 2 0.627307	Group 3 0.577274	Group 4 0.540347	Group 0 0.487571	Group 5 0.469262	Group 6 0.22306	Group 7 0.088973	Group 8 0.075843	Group 9 0.020727	2.671145506	2.8860909
31	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
32	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
33	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
34	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
35	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
36	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
37	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
38	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
39	Group 3 0.954935	Group 2 0.859037	Group 4 0.825491	Group 1 0.729075	Group 0 0.70367	Group 5 0.655239	Group 6 0.381012	Group 7 0.145112	Group 8 0.084808	Group 9 0.03241	2.869529652	2.495541555
40	Group 1 0.671512	Group 4 0.648659	Group 3 0.516097	Group 2 0.439995	Group 1 0.420614	Group 6 0.414337	Group 0 0.286113	Group 7 0.126287	Group 8 0.216205	Group 9 0.07445	2.869529652	2.968128343
41	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
42	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
43	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
44	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
45	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
46	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053
47	Group 4 0.958762	Group 5 0.90523	Group 3 0.826455	Group 6 0.674819	Group 2 0.592103	Group 1 0.458042	Group 0 0.405212	Group 7 0.350459	Group 8 0.233562	Group 9 0.110596	3.052649696	2.598160053

Figure 43: More Ambiguity in Top-Level Node Inference of Bottom-up Evidence

For instance, looking at inference performance on $t \in [29,47]$, we find that the distribution over Markov-chains indicates more ambiguity now (Figure 43) in its recognition of bottom-up evidence than it did with un-transformed data (Figure 40). Second, when considering how values of $\max\left[\lambda_t(g_r)\right]$ change from $t \in \{0,10,20, \ldots 90\}$ to $t \in \{1,11,21, \ldots 91\}$, we see interesting results. For instance, for $t \in [10,11]$, $\max\left[\lambda_t(g_r)\right]$ of the top-level node indicates *g1*. But, for $t \in [20,21]$, $\max\left[\lambda_t(g_r)\right]$ of the top-level node indicates *g1* and then *g2*. This tells us that the network recognizes the shocked flow as being closer to the equilibrium flow sometimes (e.g., $t \in [10,11]$) and different from it at other times (e.g., $t \in [20,21]$). This can be seen in Figure 44.

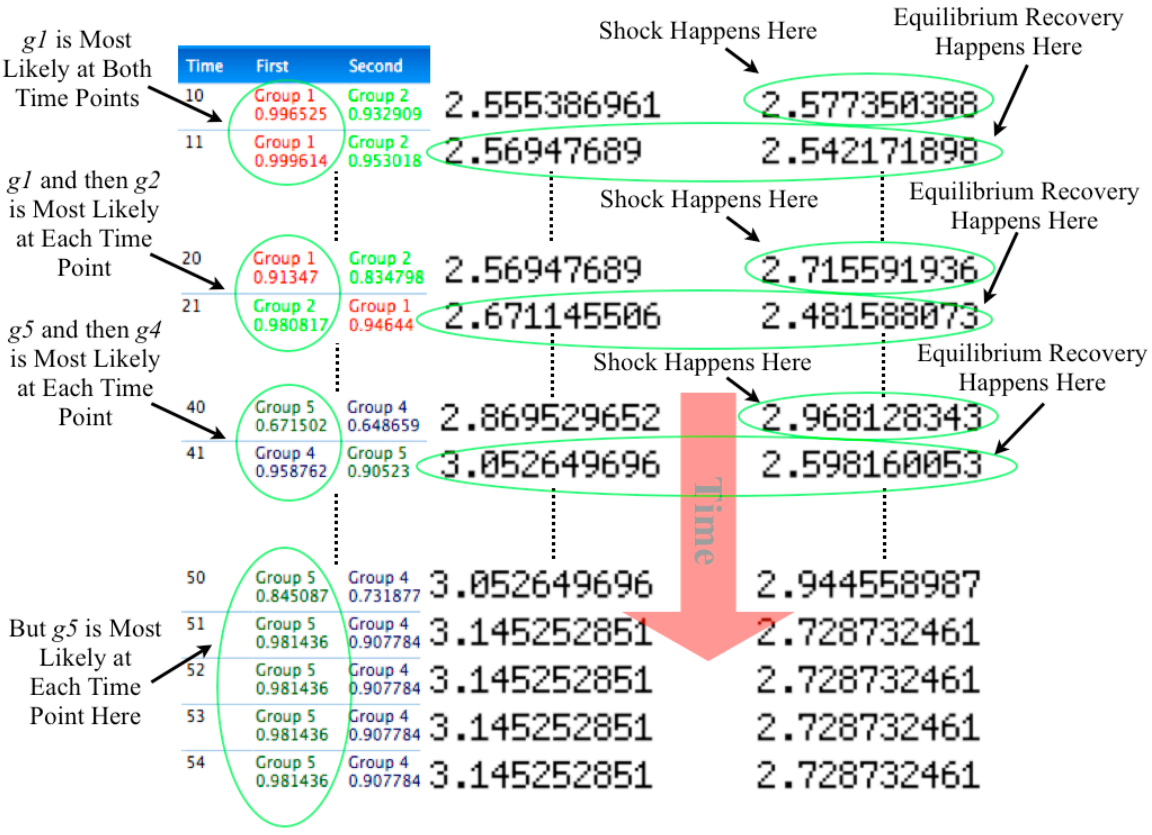


Figure 44: Imperfect Flow Recognition at Transitions

As the figure shows, the recognition is even more confused at later times. For instance, at $t \in [40,41]$, $\max\left[\lambda_t(g_r)\right]$ of the top-level node indicates *g5* and then *g4*, where *g5* is the value indicated by $\max\left[\lambda_t(g_r)\right]$ during $t \in [50,60]$. Similar flaws are seen at comparable times later in the inference history. What then can we conclude then about this network's Level 2 SA thus far?

Though the Level 2 SA of this network is somewhat flawed, it comes closer to our goals for this SA. Recall that one goal is to recognize patterns in the flow from the properties. Another goal is to recognize shock conditions. Whereas the unsupervised network trained on un-transformed data seemed to falter, the one trained on transformed data seems to come closer to each of these goals. When this network performs inference on perturbed log-transformed data, the results are commendable as well. This indicates that bottom-level nodes are more flexible in their response to non-training data. But since the inference on the original training data was not satisfactory, we will not present these results. Nevertheless, we know that the bottom-level nodes are more effective when the data is logarithm-transformed. So let us now add supervision to the training process. This will allow us to see if the log-transformation of the data can improve recognition performance of the perturbed data. Of course, this implies that we are switching goals now back to the recognition of shocked/un-shocked flow.

Knowing that inference on training data is flawless for the supervised network, let us turn immediately to its inference on perturbed data subjected to the log-transformation. The complete network parameters for this network can be found in an appendix (B.11). But the key differences to point out are that the top-level node's supervised learning algorithm is *maxProp* and the bottom-level node's *maxDistance* value is *0.0*. The former choice (*maxProp*) is to employ more of a winner-take-all approach in the top-level node's reading of bottom-up inputs. The latter choice (*maxDistance*) is done to account for the

smaller changes in bottom-up evidence values. Looking at $t \in [50,68]$ again, we see a change in inference performance from the un-transformed case (Figure 45).

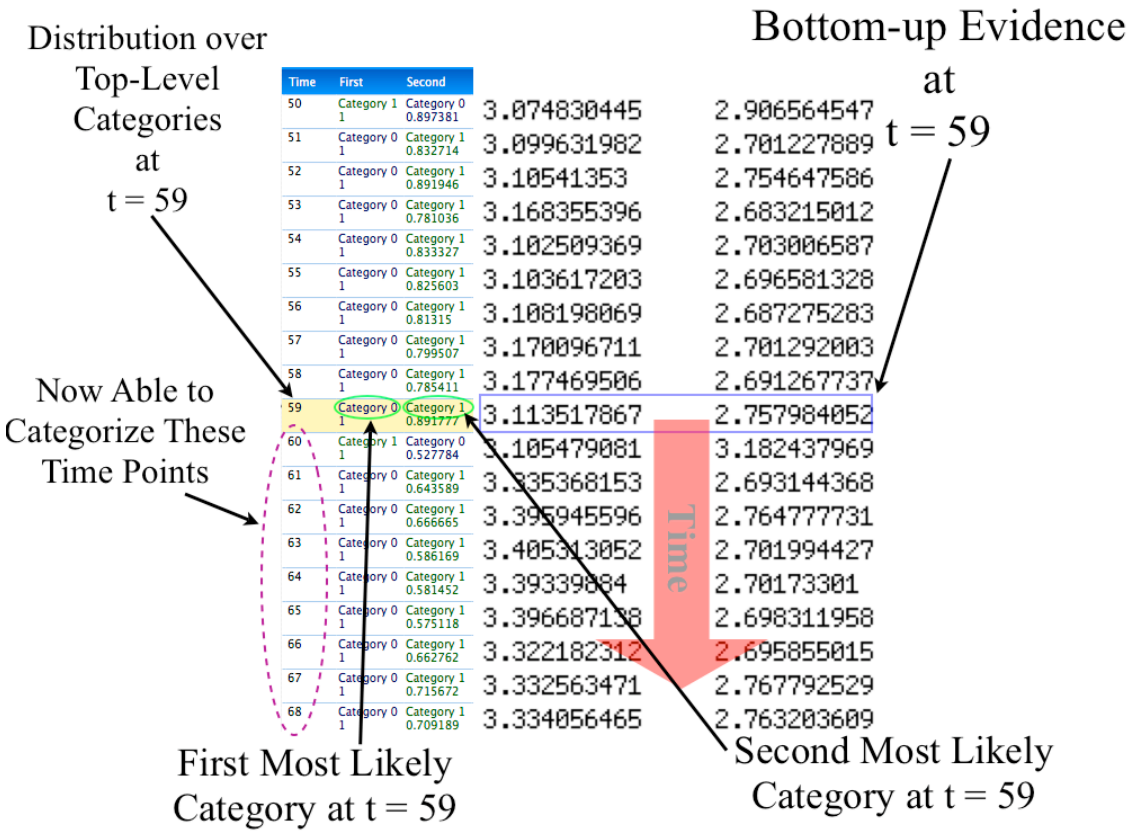


Figure 45: Supervised Network More Robust to Perturbations of Log-Transformed Evidence

The same consistent recognition performance seen in Figure 45 continues for $t \in [69,99]$. Now, the network flawlessly recognizes shocked and un-shocked flow during this time period. This can be seen from the values of $\max\left[\lambda_t(g_r)\right]$ at each time point. This is because bottom-level nodes are more able to recognize perturbations to the data. Consequently, this recognition propagates up to the top-level as feed-forward input ($\lambda^{1,1}$, $\lambda^{1,2}$). Also, we see from looking at the distribution over Markov-chains of the top-level node ($G^{2,1}$) that it is more spread out as well. So we see from this experiment that an HTM trained on log-transformed data of T and u can learn to differentiate shocked from

un-shocked flow. This conclusion assumes though that the dataset used as bottom-up evidence in inference does not change substantially (i.e., by more than 10%) from the training dataset.

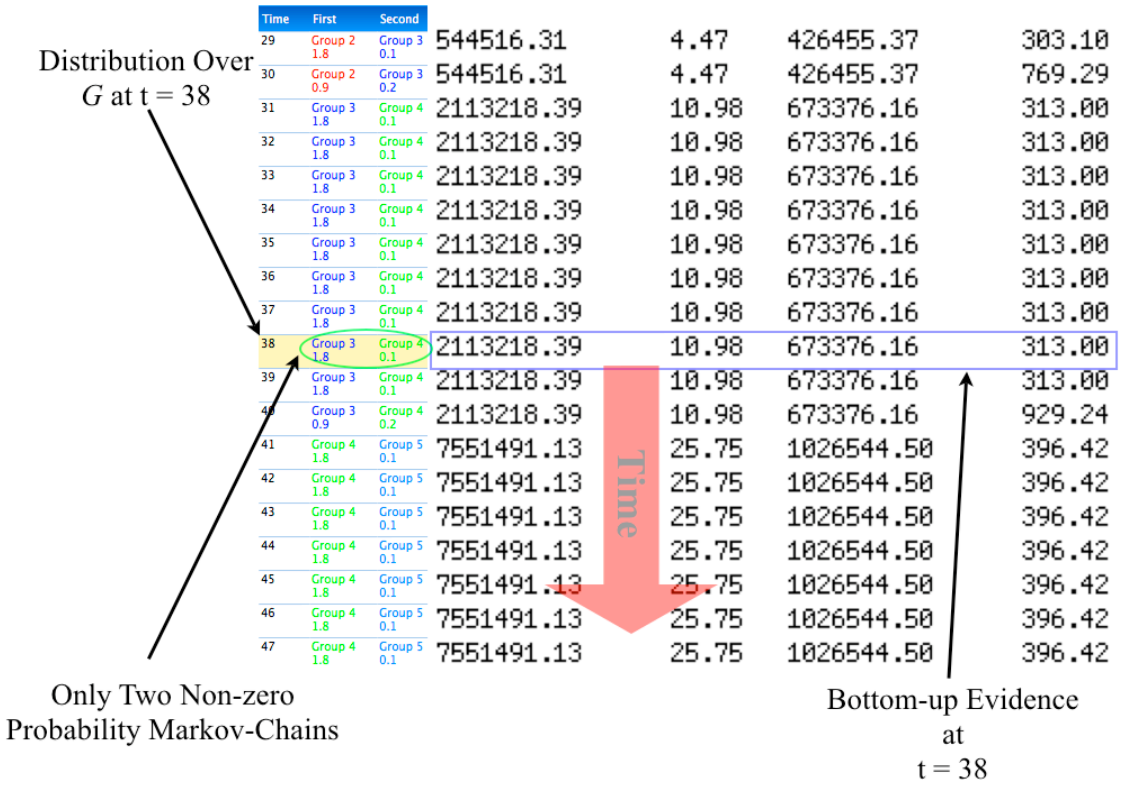
To this point, we have seen that a supervised network can exhibit commendable recognition performance of shocked and un-shocked flows, but we still need more to bridge the way to stability analysis of Iraq. First of all, we want to use unsupervised networks to extract the implicit information. Thus far, unsupervised networks have demonstrated mediocre performance in this regard with respect to both goals. We have speculated that more dimensions to the dataset might provide some support. This would be in line with data fusion principles [23]-[31] that claim that redundant observation of the same phenomenon can lead to better understanding. Thus far, supervision has provided the extra guidance to the learning process, but only when the goal is recognizing shocked flow. Of course, we have not tested the use of supervision in light of the second goal (recognizing different flow patterns). But it is conjectured that the performance would be comparable. It is believed though that more data would help accomplish this goal, instead of mapping the data-poor learning results from $N = 2$ data to *a priori* determined categories. So that is where our next major experiment in the shockwaves context takes us.

Experiment #2: Level 2 SA with Four-dimensional Data

The major focus of this experiment is to create Level 2 SA with an unsupervised network. Of course, this is in light of two goals: the first is to recognize shocked/un-shocked flow and the second is to recognize unique flow patterns. For a sanity check, we will still examine supervised network results, but we expect these to follow somewhat from both the unsupervised results and the work done in $N = 2$ above. As we will see, the log-transformation plays an important role in the accuracy of the Level 2 SA formed by an HTM network.

We proceed first to look at unsupervised learning of the $N = 4$ $V_{properties}$ vector space. An excerpt of this data is exactly what is shown in Table 6 and the complete dataset is found in an appendix (B.1). Following our evolutionary approach to network design, nearly the same network that had been used for the Waves context ($V_{property}$) is used here. Thus far, results on $N = 2$ training/inference have shown the extend-ability of HTM to $V_{properties}$ datasets. But the use of this network in our $N = 4$ Level 2 SA formation would more directly lend credence to that extension. The complete algorithm parameters for this network are shown in an appendix (B.12). In short, the network implemented here has three levels, rather than the four used in the Waves example. This is done because of the lower dimensionality of the vector space that led to our conjecture that an additional level of processing was not needed. After training the network, we form ten top-level Markov-chains ($G^{3,1}$) and nineteen coincidence patterns ($C^{3,1}$).

Now we look at the network's inference performance to gauge its Level 2 SA. As before, we begin by looking at the inference performance on the original training dataset (see appendix B.13). First, the distribution over Markov-chains in $N^{3,1}$ is non-zero for only two Markov-chains at each time step. For instance, looking at inference performance on $t \in [29,47]$, we find that the distribution over Markov-chains indicates nearly no ambiguity in its inference of bottom-up evidence (Figure 46).



Time	First	Second				
29	Group 2 1.8	Group 3 0.1	544516.31	4.47	426455.37	303.10
30	Group 2 0.9	Group 3 0.2	544516.31	4.47	426455.37	769.29
31	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
32	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
33	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
34	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
35	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
36	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
37	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
38	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
39	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
40	Group 3 0.9	Group 4 0.2	2113218.39	10.98	673376.16	929.24
41	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
42	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
43	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
44	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
45	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
46	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
47	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42

Figure 46: Unsupervised Network Inference on $N = 4$ Training Data

Second, when considering how values of $\max\left[\lambda_t(g_r)\right]$ change from $t \in \{0,10,20, \ldots 90\}$ to $t \in \{1,11,21, \ldots 91\}$, we see results that match our understanding from gas dynamics literature. For instance for $t \in [10,11]$, $\max\left[\lambda_t(g_r)\right]$ of the top-level node no longer indicates only *g1*, as it did for the corresponding $N = 2$ unsupervised implementation. Rather, at $t = 10$, $\max\left[\lambda_t(g_r)\right]$ indicates *g0* and at $t = 11$ $\max\left[\lambda_t(g_r)\right]$ indicates *g1*. Similarly for $t \in [20,21]$, $\max\left[\lambda_t(g_r)\right]$ of the top-level node indicates *g1* and then *g2*. This can be seen, as well as what is expected for $t \in \{[40,41], [50,54]\}$, in Figure 47.

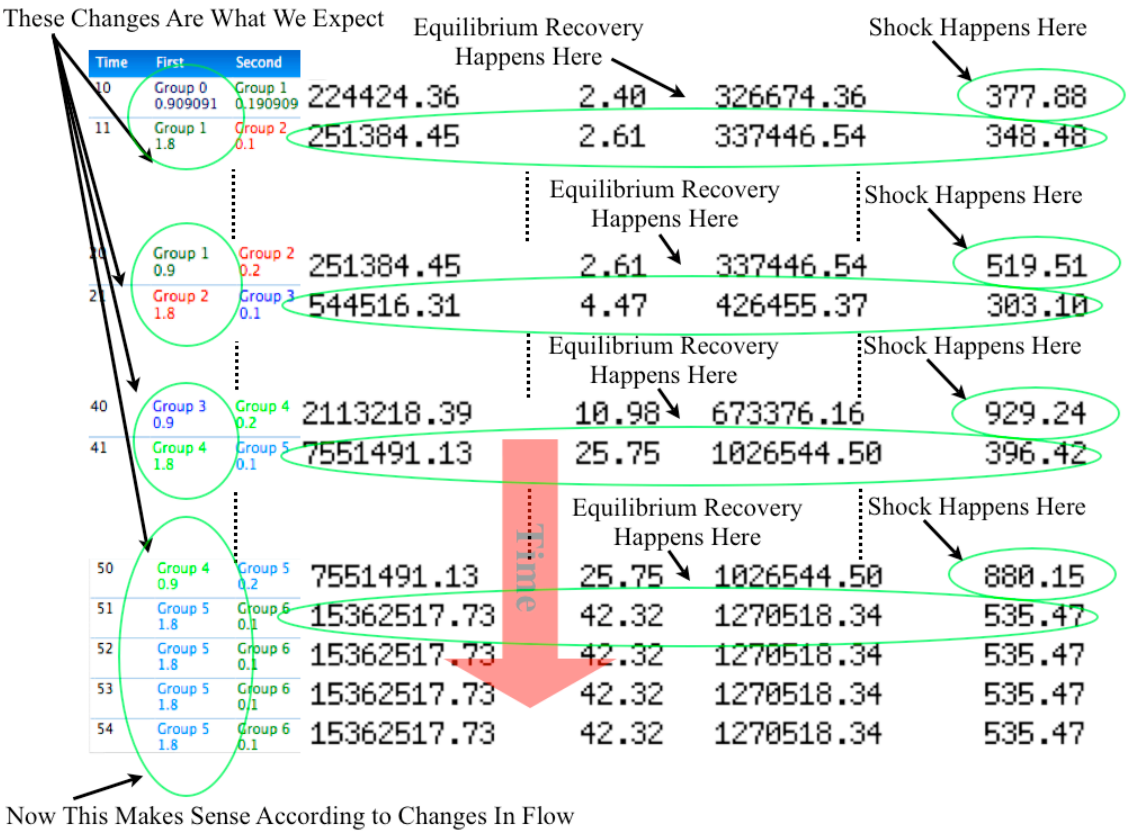


Figure 47: Expected Flow Recognition at Transitions

This trend in $\max\left[\lambda_t(g_r)\right]$ continued for similar transitions from $t \in \{30, 40, \ldots 90\}$ to $t \in \{31,41, \ldots 91\}$. Consequently, we know that the network recognizes the shocked flow as being closer to the equilibrium flow that *precedes* it each time. Recall earlier that it more often recognized the shocked flow as being closer (via $\max\left[\lambda_t(g_r)\right]$) to the equilibrium flow that *followed* it. So we seem to have scored a small victory in Level 2 SA here. In short, this network recognizes that flow with a shock in it is more similar to the flow without a shock, and much more different than the flow to follow. But how has it done this?

If we look at Euclidean distances between these vectors, then we see why this is so. If we take the Euclidean distance between vectors at $t \in \{9,19, \ldots 89\}$ and $t \in \{10,20,$

...*90}*, as well as vectors at $t \in \{10,20, ...90\}$ and $t \in \{11,21, ...91\}$ (Table 8), then we see that this victory should be expected.

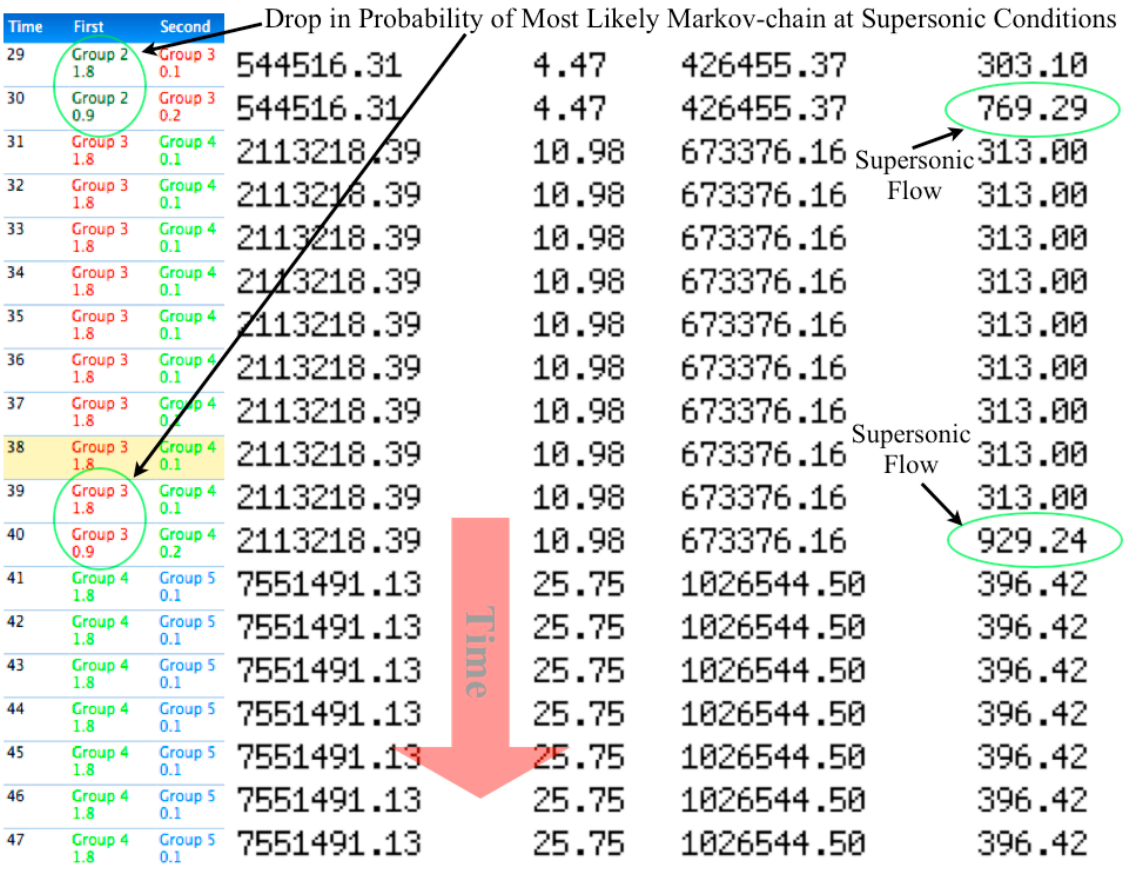
Table 8: Euclidean Distances Between All Properties at Flow Transitions

time	distance
time step	unitless
9	124.685716
10	29032.5218
11	0
19	171.0324194
20	306347.6868
21	0
29	466.1899484
30	1588016.471
31	0
39	616.2429617
40	5449728.3
41	0
49	483.7305452
50	7814835.884
51	0
59	863.183469
60	51073930.49
61	0
69	1181.357749
70	193935998
71	0
79	525.7044914
80	37294243.08
81	0
89	1187.057332
90	972612402.3
91	0

From literature on HTM inference algorithms [119] we know that this factors into the calculation of $\lambda_t(g_r) = P(e_t \mid g_r)$ at each value of t. From the Euclidean distances, we see that the changes due to the shock are far greater than those of the post-shock recovery. This is because of the huge changes in properties from the shock. Consequently, in terms of a machine-learning algorithm, the top-level node's feed-forward output is not much of

an improvement over a simpler distance calculation. Nevertheless, for our goal of recognizing flow patterns, it seems that this Level 2 SA is satisfactory.

But there is another victory in terms of Level 2 SA for the other goal. It is that the network allows its user to recognize – though, non-uniquely – a shock as distinct from the flow that follows. Consequently, a user can then recognize the difference between shocked and un-shocked flow from this feed-forward inference. We can see this by looking at the distribution over top-level Markov-chains for $t \in [29,47]$ (Figure 48).



Time	First	Second				
29	Group 2 1.8	Group 3 0.1	544516.31	4.47	426455.37	303.10
30	Group 2 0.9	Group 3 0.2	544516.31	4.47	426455.37	769.29
31	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
32	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
33	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
34	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
35	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
36	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
37	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
38	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
39	Group 3 1.8	Group 4 0.1	2113218.39	10.98	673376.16	313.00
40	Group 3 0.9	Group 4 0.2	2113218.39	10.98	673376.16	929.24
41	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
42	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
43	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
44	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
45	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
46	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42
47	Group 4 1.8	Group 5 0.1	7551491.13	25.75	1026544.50	396.42

Figure 48: Non-unique Recognition of Supersonic Conditions with Unsupervised Network

We see here that for $t = 40$ the value of $\max\left[\lambda_t(g_r)\right]$ is lower than what it is for $t \in \{31,39\}$, though the indicated group number is still the same. This drop in probability happens for each $t \in \{10,20, \ldots 90\}$ in comparison to each $t \in \{1\text{-}9,11\text{-}19, \ldots 81\text{-}89\}$. We

could interpret this consistency to indicate that the network recognizes something happening at these time values. The values are also relatively high, in comparison to what they were earlier with the $N = 2$ corresponding experiment. So we see here from looking at the network's feed-forward inference that the data is telling us something at each of these junctures. These junctures are related via their relative drop in probability values. Consequently, we can say that the network allows us – the human users – to non-uniquely identify something implicit in the data at each of these points. Of course, we know *a priori* from the modeling that generated this data that each instance is actually a shock. But for cases when this *a priori* knowledge is not available, this Level 2 SA could be highly valuable.

Given these apparent victories, let us look though at the inference abilities of this network for a host of testing datasets. This will further allow us to test the Level 2 SA formed with this unsupervised network (see network parameters in appendix B.12). We begin rather plaintively by picking a constant noise factor to the original training data that is on the order of the bottom-level *maxDistance* parameter. Consequently, a perturbation of the data by only *+0.04* is done initially. Recall that the value of *maxDistance* for this network is *0.05*. Upon inference of this dataset, the network clearly recognized each flow type as it had done with the original training data. The same results are obtained by a perturbation of *-0.04*, indicating that these miniscule perturbations have no effect on inference. Physically speaking, these perturbations amount to changes of 0.000000003149% (e.g., for $P(t = 99)$) at the smallest, and 0.0097% (for $u(t = 0)$) at the largest. In other words, these perturbations are nowhere near the magnitude tested earlier. So while these perturbations are comparable to the value of *maxDistance* used, they do not just yet tell us much about the physical relevance of this Level 2 SA.

So let our perturbation grossly exceed the value of *maxDistance* and see what the effect is on the inference. We pick a perturbation of *10* to the training data. At the least this would create a 0.000000787% change (e.g., for $P(t = 99)$) and at the most it would

create a 676.59% change (for $\rho(t = 0)$). This would be somewhat of an unphysical perturbation to the data because of the huge perturbation to ρ. So we see here a problem if property values are left in SI units. Perhaps it would be better in the future to have all properties on a comparable scale. For now though, we will keep SI units to assess what, if any, problems arise from keeping this relative scale between the properties' values. If we leave ρ unperturbed then the greatest perturbation is a 2.37% change (for $u(t = 0)$). So this is how we perturb the data. *P, u* and *h* are perturbed by a value of *10* and ρ is left as it was during training. When tasked with inference on this dataset, the bottom-level node over the *u* sensor ($N^{1,4}$) outputs a zero probability distribution for $\lambda_t(g_r)$ nearly $\forall t \in [0,99]$. This output then zeros the output of its parent node ($N^{2,2}$). But the top-level node resolves the ambiguity by using the output of the other middle-level node ($N^{2,1}$), which is barely changed (Figure 49).

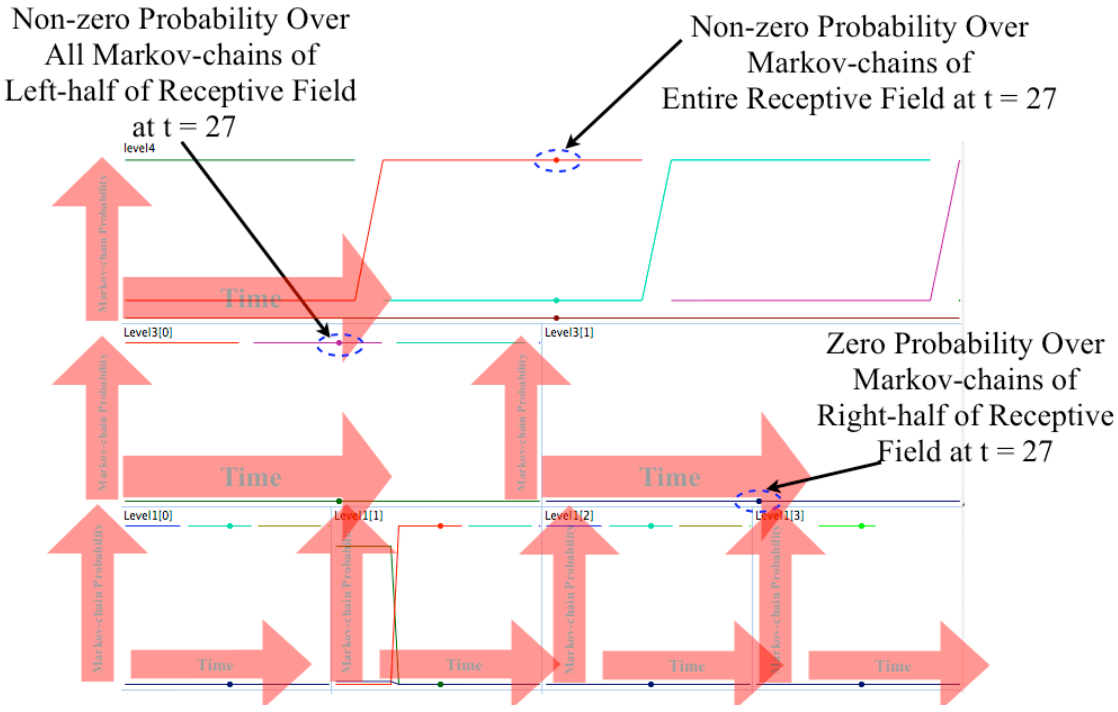


Figure 49: Top-Level Node Resolves Ambiguity of Second-Level Nodes' Output

Consequently, the top-level output is still in line with what we have seen for previous perturbations. But now we see feed-forward inference unraveling in the intermediate levels of the network. So we see a fundamental flaw in our apparent victory from earlier. While this network satisfies our goal of recognizing certain flow patterns, its performance starts falling off as the training data is perturbed by at most 2.37%. If we had perturbed the data in the left half of the receptive field (i.e. values of ρ and P) by this percentage then the $N^{2,1}$ node's $\lambda_t(g_r)$ would also be have been zeroed out. This would then propagate up to the top-level, causing its $\lambda_t(g_r)$ to be zeroed out as well. So we see a problem in the making here. The network will not recognize perturbations to data across its entire receptive field. Furthermore, the tentative satisfaction of our other goal of recognizing shocks in the flow will be stymied as well. Consequently, the Level 2 SA of this network would be unsuitable. But would adding supervision help matters again as it did for the corresponding $N = 2$ experiment?

So we now train a supervised network, providing in addition to the $N = 4$ $V_{properties}$ the category data. The parameters for this network can be found in an appendix (B.12), but the main difference is that the top-level node is now a supervised node. After training, the top-level node learned nineteen coincidence patterns ($C^{3,1}$) and mapped its Markov-chains to the two categories of interest. Since this network is built on the unsupervised learning of 'Level 1' and 'Level 2', it is expected that faults in those level's feed-forward inference should propagate up to the top-level node. But will the top-level node resolve the ambiguity as it did for the unsupervised network?

Inference tests on this supervised network reveal a surprising insight. There was no change in performance from the unsupervised network, except when the perturbation of *10* was used. This means that the supervised network is unable to resolve the ambiguity of the 'Level 1' and 'Level 2' nodes in this case. So with a different goal for the network (i.e., to recognize shocked/un-shocked flow), we see how the resulting SA

has changed. It is unable to recognize shocked flow when the perturbation in part of the receptive field is at most 2.37% from what it trained on.

We can summarize our findings from each of these perturbation tests. Inference of the training data resulted in flawless recognition as expected. The miniscule perturbations in either direction also created no change in recognition. The perturbation by *10* proved too large for top-level node. When this bottom-up evidence was in the receptive field, the probability distribution over the 'Level 1' and 'Level 2' Markov-chains was completely zero. Then the top-level node could not map this feed-forward inference from $N^{2,2}$ to learned categories. So while supervision helped matters in the $N = 2$ case of un-transformed data, it has absolutely no effect here in the $N = 4$ case. The root of this problem is the $N^{1,4}$ node's calculation of $\lambda_t(g_r) = P(e_t \mid g_r)$ being zero nearly $\forall t$.

It might seem logical then to change the inference algorithm to one more suitable to the changes in the data. But the Gaussian inference algorithm used here depends on distance calculations between bottom-up evidence and stored patterns in C. If this inference algorithm is to be used then it should not respond to changes of 2.37% for a property. A change of at most 2.37% is not a significant change and we want the network to contain this understanding in its knowledge base. But for this network, the differences between the perturbed data (i.e., $\tilde{e}_t$) and the vectors stored in C of $N^{1,4}$ are too great. This is the node over u. Consequently, since $\lambda_t(g_r) = P(e_t \mid g_r)$ is calculated with the Gaussian inference algorithm, $\lambda_t(g_r)$ is almost always zero for one out of four of the bottom-level nodes. These problems would be alleviated though if both the training and testing data were transformed somehow. So let us proceed directly from these experiments to the learning/inference of log-transformed data. This may alleviate the problems seen when applying a perturbation across the dataset. Furthermore, we might then be able to apply the perturbation to *all* of the properties, rather than just those for which it is physically reasonable.

So we transform the $V_{properties}$ with a logarithm and then use this data for training an HTM network. First, we do this for an unsupervised network and then for a supervised network. The data used for training can be found in an appendix (B.15). The supervision data used for training is the same as it was in previous cases, i.e. a *1* to indicate supersonic and a *0* to indicate subsonic conditions.

We first train an unsupervised network on this log-transformed data and assess its inference capabilities. The unsupervised network used here is a modified version of what was used in the un-transformed case. The network parameters can be found in an appendix (B.16), but the main point is that the network has only two levels. This is done because preliminary testing led to the conclusion that the third level (used previously) was extraneous. This was not the reason though for the previous network's failure. Rather, it is believed that the bottom-level's calculation of $\lambda_t(g_r) = P(e_t \mid g_r)$ being zero over a large enough perturbation caused this failure. Nevertheless, the third level of the network is believed to be extraneous and so we have removed it. When the two-level network's top-level node learned twenty coincidence patterns ($C^{2,1}$) and ten Markov-chains ($G^{2,1}$), it was clear that this conjecture was correct. This is because we know *a priori* about the ten flow patterns from the simulation that created the data.

We can see directly from one of the bottom-level parameters how this learning happened. We note that the value of *maxDistance* used here was *0.0*. This parameter choice causes three of four bottom-level nodes (those above P, ρ, h sensors) to learn eleven patterns. Each pattern corresponds to the initial condition plus ten post-shock equilibrium conditions, making eleven in total. But the fourth bottom-level node (above the u sensor) learned only nineteen patterns. Why? We would expect it to have learned twenty because of the ten shocks and the ten subsequent equilibrium recoveries, making twenty in total. It is suspected that the sparse representation of the coincidence matrix ($C^{1,4}$) caused the first shock to be considered non-distinct from the equilibrium flow that followed. This may have caused nineteen, rather than twenty, coincidence patterns to be

learned in this node. With this narrow margin of error for bottom-level spatial pooling, the top-level node was able to learn ten distinct Markov-chains. We must now turn to inference testing to see if these Markov-chains correspond to the flow patterns we know to exist.

We first look at inference on the training data. So as before, we examine values of $\max\left[\lambda_t(g_r)\right]$ for each *t* (see appendix B.18 for complete history). In doing so, we see that the change in most likely Markov-chain from $t \in \{20, 30, 40, 50, 60, 70, 90\}$ in comparison to each $t \in \{21, 31, 41, 51, 61, 71, 91\}$ is as we expect. Specifically, the change in most active Markov-chain for $t \in \{21, 31, 41, 51, 61, 71, 91\}$ from what it was for $t \in \{20, 30, 40, 50, 60, 70, 90\}$ corresponds exactly to what we expect for flow type recognition. The network clearly shows with observations of these points that the network recognizes seven equilibrium flow types. The only faults are in $\max\left[\lambda_t(g_r)\right]$ at $t \in \{0,1\}$, $t \in \{10,11\}$ and $t \in \{80,81\}$. We will consider the first two, since the third occurs for similar reasons as the second. Looking at the first fault, it is not clear why this occurs. It is likely due to the sparse representation of $C^{1,4}$, causing $\bar{e}_0$ not to appear distinct enough from $\bar{e}_1$. When this feed-forward inference propagates up the network, the flow at $t = 0$ looks similar enough to the flow at $t = 1$. Let us look at the second fault now. Examination of the probability distribution over Markov-chains reveals that the difference in first and second most probable Markov-chains is small (*~0.1*, as opposed to *~0.4* for subsequent comparable transitions). This indicates some ambiguity in the network's inference of the evidence. So for seven out of ten cases, it seems that Level 2 SA of this network is suitable (i.e., 70% accuracy), having considered only the training data thus far. Of course, in the scope of all time points, these are three errors out of one hundred points. But since the bottom-up evidence does not change for 70% of the time history, we will not claim a higher accuracy rate just yet.

Before turning to perturbation testing, we should also note that the network's feed-forward output allows the human users to see that something happens each time the shock occurs. Specifically, each time there is a shock in the flow, the value of $\max\left[\lambda_t(g_r)\right]$ drops from what it had been for the previous nine time steps, ignoring the *t = 0, t = 10* and the *t = 80* shocks. Even the *t = {0,10,80}* shocks reflect a drop in the value of $\max\left[\lambda_t(g_r)\right]$ from the immediate vicinity in time. As before, this allows us the users to see – with 70% accuracy – something is happening here that is somewhat of a perturbation from the situation that preceded it for the previous nine time steps. Of course, as was said before, this could have been recognized also with a Euclidean distance calculation between vectors of successive time steps.

In our inference testing, we probe the ability of this network to recognize the ten flow patterns it has learned from its training. We realize from the un-transformed implementation that perturbations cannot be chosen based only on the network parameters. Rather, for the resulting Level 2 SA to be of any physical use, the perturbations must be physically relevant. Consequently, to test the inference we jump immediately to a 5-10% random perturbation of the original training data in which energy (via $h \propto T$) is held constant. An example dataset resulting first from this perturbation and then from the log-transformation is shown in an appendix (B.17). It is important not to reverse the order of the perturbation and the transformation. Once we have this data, it is possible to test inference performance. A complete history of inference performance is also shown in an appendix (B.19). There are three noticeable differences from inference on the original training data. The first is at *t = 80*, where instead of $\max\left[\lambda_t(g_r)\right]$ being for *g7*, it is for *g8*. Including this error with the comparable one at *t = 10* (already done on the training data), and the other at *t = 0*, this makes for seven out of ten times that the network recognizes the shock as being more related to the preceding equilibrium flow

than the one to follow. Aside from these three time steps, the network recognizes the ten flow patterns via the value of $\max\left[\lambda_t(g_r)\right]$ quite well (97% accuracy).

It is interesting to see how the perturbations affect the top-level's distribution over Markov-chains. For instance, we can visually compare the distribution for the 5-10% perturbed case to that of the training data (Figure 50).

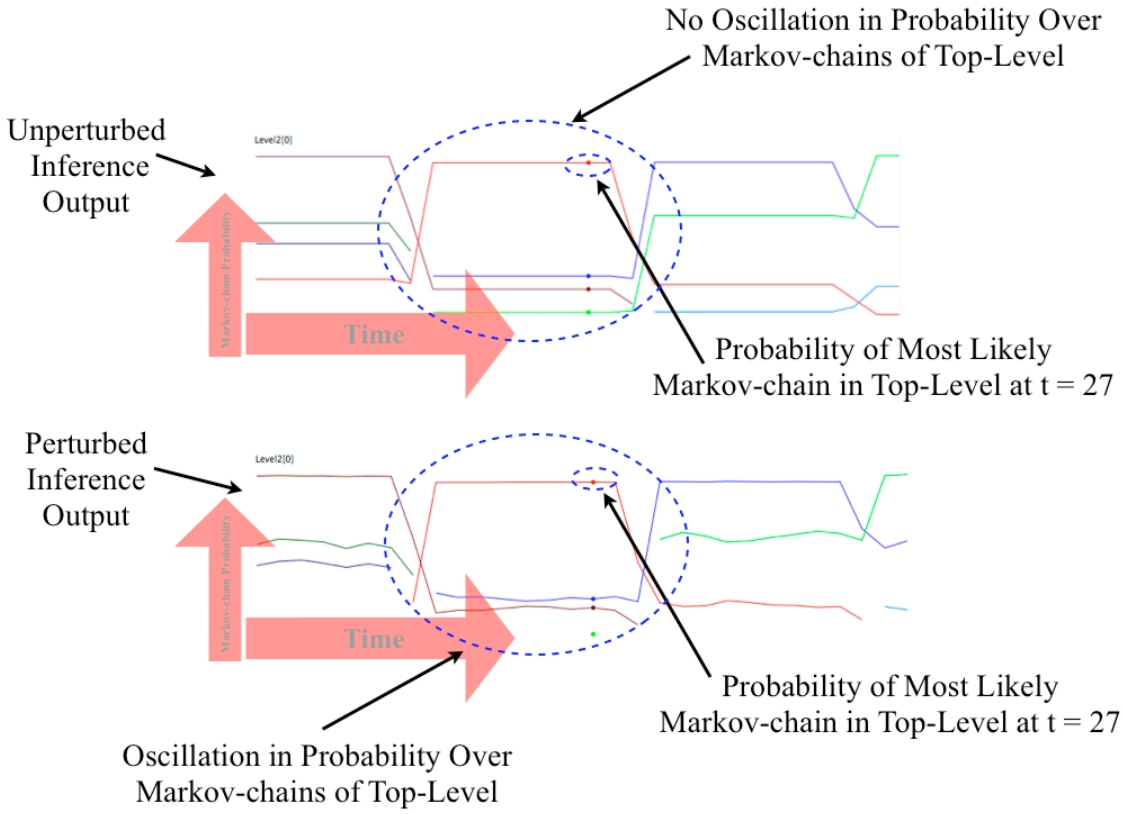
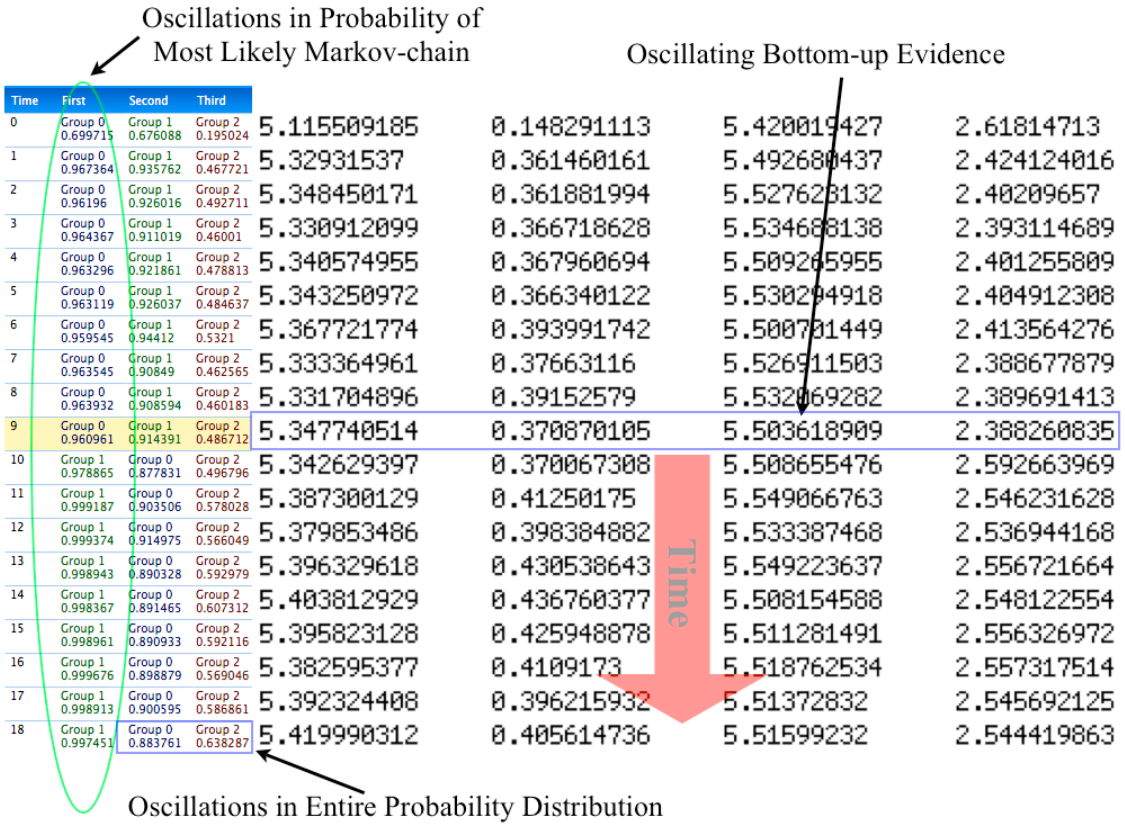


Figure 50: Effects of Perturbed Data in Top-Level Feed-forward Inference

The oscillation is most pronounced over non-maximum values of $\lambda_t(g_r)$ coming from $N^{2,1}$. Examination of the actual inference output (see an excerpt in Figure 51) shows that $\max\left[\lambda_t(g_r)\right]$ also oscillates slightly in response to the significant oscillation in the bottom-up evidence.



Time	First	Second	Third				
0	Group 0 0.699715	Group 1 0.676088	Group 2 0.195024	5.115509185	0.148291113	5.420019427	2.61814713
1	Group 0 0.967364	Group 1 0.935762	Group 2 0.467721	5.32931537	0.361460161	5.492680437	2.424124016
2	Group 0 0.96196	Group 1 0.926016	Group 2 0.492711	5.348450171	0.361881994	5.527628132	2.40209657
3	Group 0 0.964367	Group 1 0.911019	Group 2 0.46001	5.330912099	0.366718628	5.534688138	2.393114689
4	Group 0 0.963296	Group 1 0.921861	Group 2 0.478813	5.340574955	0.367960694	5.509265955	2.401255809
5	Group 0 0.963119	Group 1 0.926037	Group 2 0.484637	5.343250972	0.366340122	5.530294918	2.404912308
6	Group 0 0.959545	Group 1 0.94412	Group 2 0.5321	5.367721774	0.393991742	5.500701449	2.413564276
7	Group 0 0.963545	Group 1 0.90849	Group 2 0.462565	5.333364961	0.37663116	5.526911503	2.388677879
8	Group 0 0.963932	Group 1 0.908594	Group 2 0.460183	5.331704896	0.39152579	5.532869282	2.389691413
9	Group 0 0.960961	Group 1 0.914391	Group 2 0.486712	5.347740514	0.370870105	5.503618909	2.388260835
10	Group 1 0.978865	Group 0 0.877831	Group 2 0.496796	5.342629397	0.370067308	5.508655476	2.592663969
11	Group 1 0.999187	Group 0 0.903506	Group 2 0.578028	5.387300129	0.41250175	5.549066763	2.546231628
12	Group 1 0.999374	Group 0 0.914975	Group 2 0.566049	5.379853486	0.398384882	5.533387468	2.536944168
13	Group 1 0.998943	Group 0 0.890328	Group 2 0.592979	5.396329618	0.430538643	5.549223637	2.556721664
14	Group 1 0.998367	Group 0 0.891465	Group 2 0.607312	5.403812929	0.436760377	5.508154588	2.548122554
15	Group 1 0.998961	Group 0 0.890933	Group 2 0.592116	5.395823128	0.425948878	5.511281491	2.556326972
16	Group 1 0.999676	Group 0 0.898879	Group 2 0.569046	5.382595377	0.4109173	5.518762534	2.557317514
17	Group 1 0.998913	Group 0 0.900595	Group 2 0.586861	5.392324408	0.396215932	5.51372832	2.545692125
18	Group 1 0.997451	Group 0 0.883761	Group 2 0.638287	5.419990312	0.405614736	5.51599232	2.544419863

Figure 51: Oscillations in Markov-chain Probabilities Due to Perturbations in Bottom-up Evidence

What is demonstrated in Figure 51 is seen $\forall t$ during inference of the perturbation dataset (see B.19). Nevertheless, the network remains able to recognize these flow patterns (via $\max\left[\lambda_r\left(g_r\right)\right]$) with high accuracy despite these oscillations.

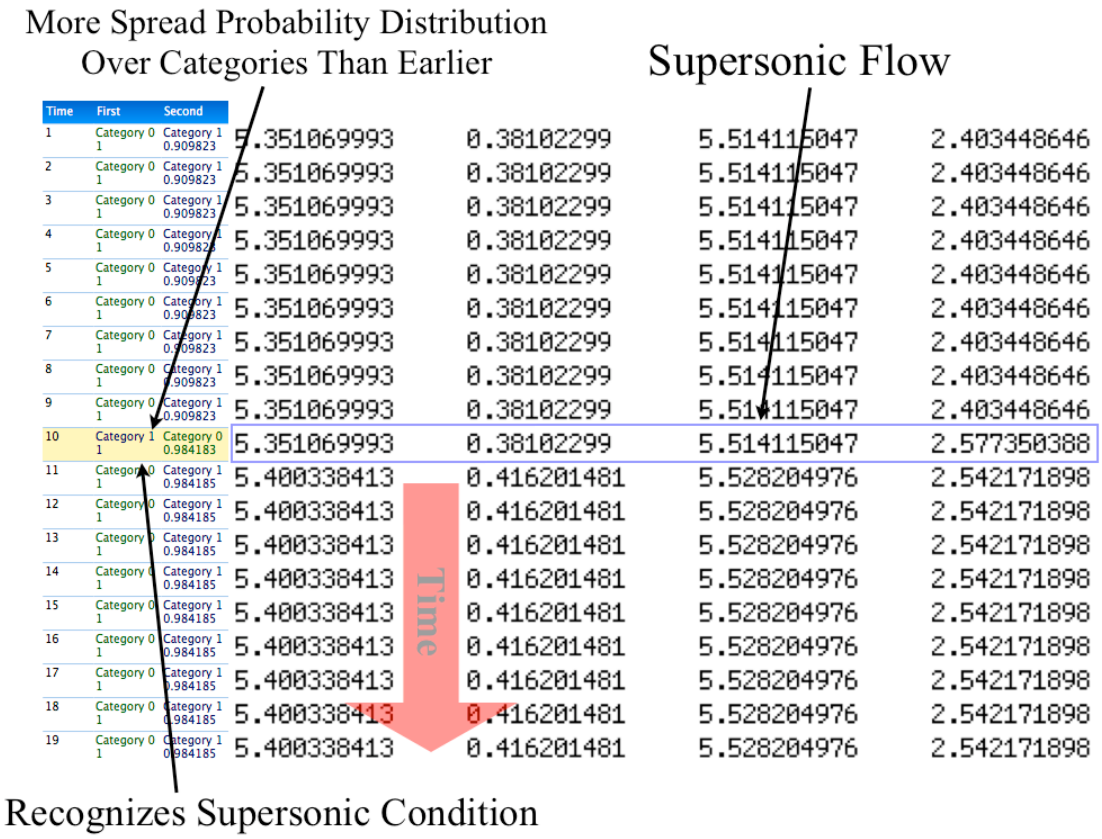
One consideration though in creating the perturbation data has been its physical consistency. For this perturbation data, energy (via h) was held constant. This was done because any property in equilibrium is a function of two others. So if the perturbation were applied to all properties without this constraint, then it is possible that oscillations in two properties could amount to more oscillation than actually occurred in the one dependent on them. For instance, if we consider enthalpy (h) then $h(P,\rho)$ implies that perturbations in h could be different from $h_{calculated}$. Specifically, we could have that $\Delta h \neq h\left(\Delta P,\Delta\rho\right)$. Doing the perturbations this way yields an average difference of 4.13%

between the perturbed and the calculated enthalpy. Even though this is a relatively small amount, it simulates one of two real-life analogs. The first one is that there is an artificial factor on the shockwaves context by which energy can escape/enter the context. If this were true then, in effect, inference performance on this dataset would not be equivalent to inference on the shockwaves context *per se*. Rather, this would be a context *similar* to the shockwaves one, but a context that is more like real flow. This is more like real flow because it frequently happens that energy enters/leaves a point in space due to ambient conditions. The second real-life analog would be conflicting information coming from sensors. There could be a problem with the actual sensor for enthalpy, yet measuring two other properties (here P and ρ) on which it depends provides multisensory data on h. Hall & Llinas remind us of this benefit in data fusion practices [23]: "[There is] statistical advantage gained by combining same-source data (e.g., [by] obtaining an improved estimate of a physical phenomenon via redundant observations)." So a good data fusion algorithm should be able to resolve this ambiguity. Considering these real-life possibilities, it seems that we should now test inference on a dataset with a uniform 5-10% perturbation regardless of the energy conservation condition.

So we make a 5-10% perturbed dataset for which enthalpy (h) is *not* held constant and, after log-transformation, we assess inference performance. The time history of inference can be found in an appendix (B.20). We see from it that, although the values of $\lambda_t(g_r)$ change, the trends in $\max\left[\lambda_t(g_r)\right]$ remain the same. This means that the 97% accuracy from the previous perturbation inference carries over to this one. But in this, one *additional* factor (as shown in Figure 37) has been added to the original shockwaves context. This is a compelling result for the greater issue of computational Level 2 SA. We see proof here that an HTM network trained on one dataset can generalize to another one that is similar to it. This similar dataset can be interpreted as the result of a semi-open system or a faulty sensor. Both of these are real-life possibilities. Nevertheless, the HTM-based SA is able to generalize substantially well to this new context.

Let us round out the discussion on inference performance of log-transformed data by looking at supervision results. This of course applies to Level 2 SA in light of the goal to recognize shocked/un-shocked flow. Even though this result was obtained with the $N = 2\ V_{properties}$, we will examine it for the $N = 4\ V_{properties}$. A supervised version of the two-level network used in the preceding experiment is constructed for the task. The complete algorithm parameters can be found in an appendix (B.21). In short, the top-level node employed the *maxProp* algorithm to map the learned Markov-chains to the provided category data. After training, the network learned twenty coincidence patterns ($C^{2,1}$) in the top-level node and mapped its Markov-chains ($G^{2,1}$) to the two-category supervision data.

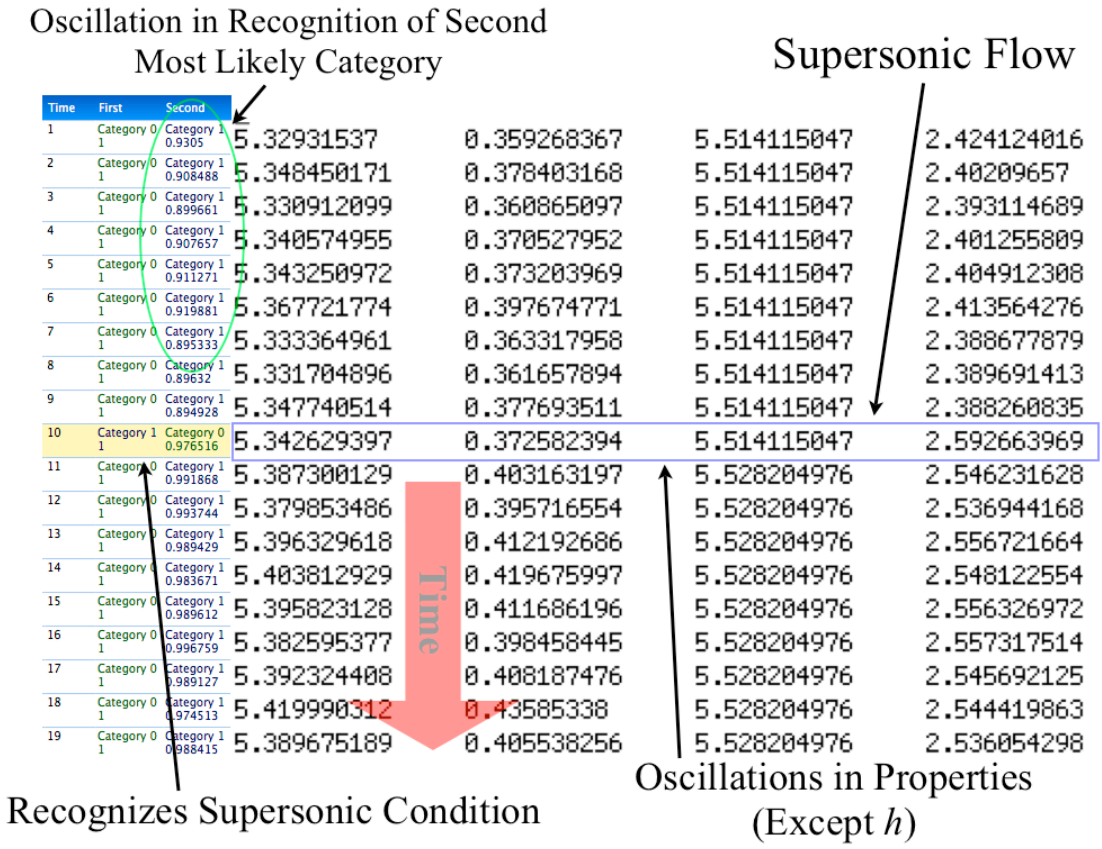
As before, inference performance on the training data does not tell us much. An example of this history can be found below (Figure 52) for $t \in [1,19]$. One thing to note from Figure 52 is that the distribution over the two categories is fairly spread out in comparison to the un-transformed analog. Nevertheless, if we follow $\max\left[\lambda_t\left(g_r\right)\right]$ as we have done thus far, we see perfect matching to shocked and un-shocked flow cases. This is true $\forall t$ of the training dataset.



Time	First	Second				
1	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
2	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
3	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
4	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
5	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
6	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
7	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
8	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
9	Category 0	Category 1	5.351069993	0.38102299	5.514115047	2.403448646
10	Category 1	Category 0	5.351069993	0.38102299	5.514115047	2.577350388
11	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
12	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
13	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
14	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
15	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
16	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
17	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
18	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898
19	Category 0	Category 1	5.400338413	0.416201481	5.528204976	2.542171898

Figure 52: Supervised Network Recognition of Log-transformed Training Data ($N = 4$)

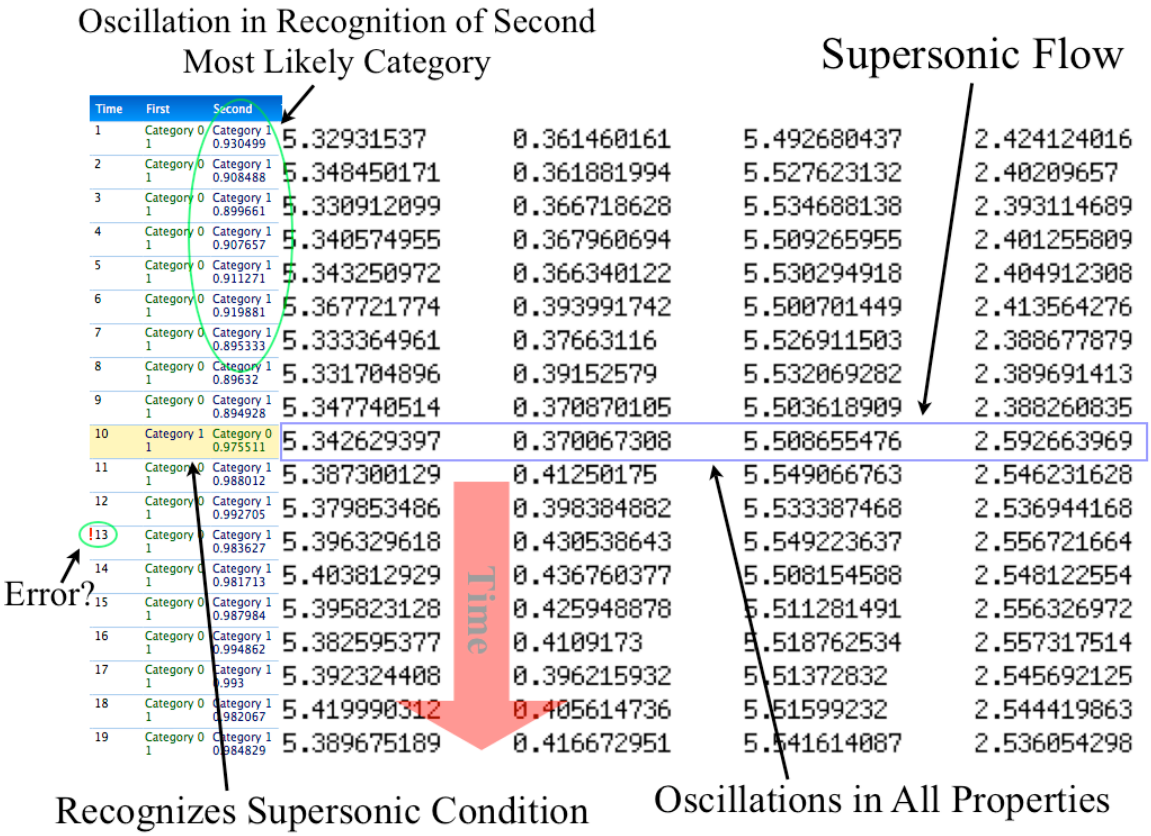
Let us probe this network in detail though by looking at inference performance on the 5-10% perturbed data. We first examine the constant *h* perturbation dataset. An excerpt of this inference history is shown below (Figure 53).



Time	First	Second				
1	Category 0 1	Category 1 0.9305	5.32931537	0.359268367	5.514115047	2.424124016
2	Category 0 1	Category 1 0.908488	5.348450171	0.378403168	5.514115047	2.40209657
3	Category 0 1	Category 1 0.899661	5.330912099	0.360865097	5.514115047	2.393114689
4	Category 0 1	Category 1 0.907657	5.340574955	0.370527952	5.514115047	2.401255809
5	Category 0 1	Category 1 0.911271	5.343250972	0.373203969	5.514115047	2.404912308
6	Category 0 1	Category 1 0.919881	5.367721774	0.397674771	5.514115047	2.413564276
7	Category 0 1	Category 1 0.895333	5.333364961	0.363317958	5.514115047	2.388677879
8	Category 0 1	Category 1 0.89632	5.331704896	0.361657894	5.514115047	2.389691413
9	Category 0 1	Category 1 0.894928	5.347740514	0.377693511	5.514115047	2.388260835
10	Category 1 1	Category 0 0.976516	5.342629397	0.372582394	5.514115047	2.592663969
11	Category 0 1	Category 1 0.991868	5.387300129	0.403163197	5.528204976	2.546231628
12	Category 0 1	Category 1 0.993744	5.379853486	0.395716554	5.528204976	2.536944168
13	Category 0 1	Category 1 0.989429	5.396329618	0.412192686	5.528204976	2.556721664
14	Category 0 1	Category 1 0.983671	5.403812929	0.419675997	5.528204976	2.548122554
15	Category 0 1	Category 1 0.989612	5.395823128	0.411686196	5.528204976	2.556326972
16	Category 0 1	Category 1 0.996759	5.382595377	0.398458445	5.528204976	2.557317514
17	Category 0 1	Category 1 0.989127	5.392324408	0.408187476	5.528204976	2.545692125
18	Category 0 1	Category 1 0.974513	5.419990312	0.43585338	5.528204976	2.544419863
19	Category 0 1	Category 1 0.988415	5.389675189	0.405538256	5.528204976	2.536054298

Figure 53: Supervised Network Recognition of Log-transformed Constant *T* Perturbed Data (*N* = *4*)

Despite slight changes in the value of the second most likely category at each time step, there is no change from the inference performance on the training data. Most likely Markov-chains indicated by $\max\left[\lambda_t(g_r)\right]$ change according to whether the flow is shocked or un-shocked, indicating the robustness of this HTM-based Level 2 SA. This continues for all values of *t* not shown in Figure 53. We then turn to inference performance on the non-constant *h* perturbation dataset. An excerpt of this inference history is also shown below (Figure 54).



Time	First	Second				
1	Category 0 1	Category 1 0.930499	5.32931537	0.361460161	5.492680437	2.424124016
2	Category 0 1	Category 1 0.908488	5.348450171	0.361881994	5.527623132	2.40209657
3	Category 0 1	Category 1 0.899661	5.330912099	0.366718628	5.534688138	2.393114689
4	Category 0 1	Category 1 0.907657	5.340574955	0.367960694	5.509265955	2.401255809
5	Category 0 1	Category 1 0.911271	5.343250972	0.366340122	5.530294918	2.404912308
6	Category 0 1	Category 1 0.919881	5.367721774	0.393991742	5.500701449	2.413564276
7	Category 0 1	Category 1 0.895333	5.333364961	0.37663116	5.526911503	2.388677879
8	Category 0 1	Category 1 0.89632	5.331704896	0.39152579	5.532069282	2.389691413
9	Category 0 1	Category 1 0.894928	5.347740514	0.370870105	5.503618909	2.388260835
10	Category 1 1	Category 0 0.975511	5.342629397	0.370067308	5.508655476	2.592663969
11	Category 0 1	Category 1 0.988012	5.387300129	0.41250175	5.549066763	2.546231628
12	Category 0 1	Category 1 0.992705	5.379853486	0.398384882	5.533387468	2.536944168
!13	Category 0 1	Category 1 0.983627	5.396329618	0.430538643	5.549223637	2.556721664
14	Category 0 1	Category 1 0.981713	5.403812929	0.436760377	5.508154588	2.548122554
15	Category 0 1	Category 1 0.987984	5.395823128	0.425948878	5.511281491	2.556326972
16	Category 0 1	Category 1 0.994862	5.382595377	0.4109173	5.518762534	2.557317514
17	Category 0 1	Category 1 0.993	5.392324408	0.396215932	5.51372832	2.545692125
18	Category 0 1	Category 1 0.982067	5.419990312	0.405614736	5.51599232	2.544419863
19	Category 0 1	Category 1 0.984829	5.389675189	0.416672951	5.541614087	2.536054298

Figure 54: Supervised Network Recognition of Log-transformed Non-Constant *T* Perturbed Data

As the table shows, besides $t = 13$, the inference is perfect. The performance at $t = 13$ is the only exception though over the rest of the inference history. This means that with 99% accuracy the network recognizes shocked/un-shocked flow in this perturbation scenario. As with the recognition of the ten flow patterns earlier in this scenario, this is a significant result because we see here the ability of an HTM to generalize to novel contexts that are likely to occur in real-life applications. But what about $t = 13$? Why was there an error there? If we examine the calculation then we can attribute it to round-off error. For example, the calculation of sound speed from $a = \sqrt{\gamma P / \rho}$ yields *359.71 m/s,* while the value of u was *360.35 m/s* at this time step. Physically speaking, these are nearly equal speeds and so the fact that the category data had indicated this point to be a

shock is attributed to round-off error. For instance, with no significant figures after the decimal we would have concluded that $u = a$ at this point. This would mean perfectly sonic conditions from which there is no discontinuous change in flow properties, i.e. the flow would still be un-shocked. So physically speaking, the inference performance on this dataset is 100% rather than 99%.

Summary of Experiments

We have seen many things from these two large experiments. We have seen that it is possible to create a commendable Level 2 SA with an HTM in the shockwaves context. For the goal of recognizing flow patterns, the unsupervised two-level network that reads $N = 4$ log-transformed data in its receptive proved the best. For the goal of recognizing shocked/un-shocked flows, both supervised networks that read log-transformed data (one that reads $N = 2$ and another that reads $N = 4$) in their receptive fields proved the best. These results have shown us some best practices for training and evaluating the inference of our HTM networks in forming SA. We have seen the merit of transforming the data in such a way that is most suitable to the learning/inference algorithms.

Also, we have seen the merit of choosing testing scenarios based on their potential for generalization within the context of interest. Such an approach is opposed to one that bases such decisions only on the algorithm parameters. As we recall, the miniscule perturbation of *0.04* was an example of this. We must remember though that the parameters are only chosen to meet a predefined goal of our SA. So testing must be done with the goals of the resulting SA in mind. This reasoning led us to the 5-10% perturbation cases to test inference, rather than perturbing the data by values comparable to *maxDistance*.

Level 3 SA and Where Things Stand

Considering the information flow from the previous chapter (reprinted below for convenience), we can evaluate what has been done thus far.

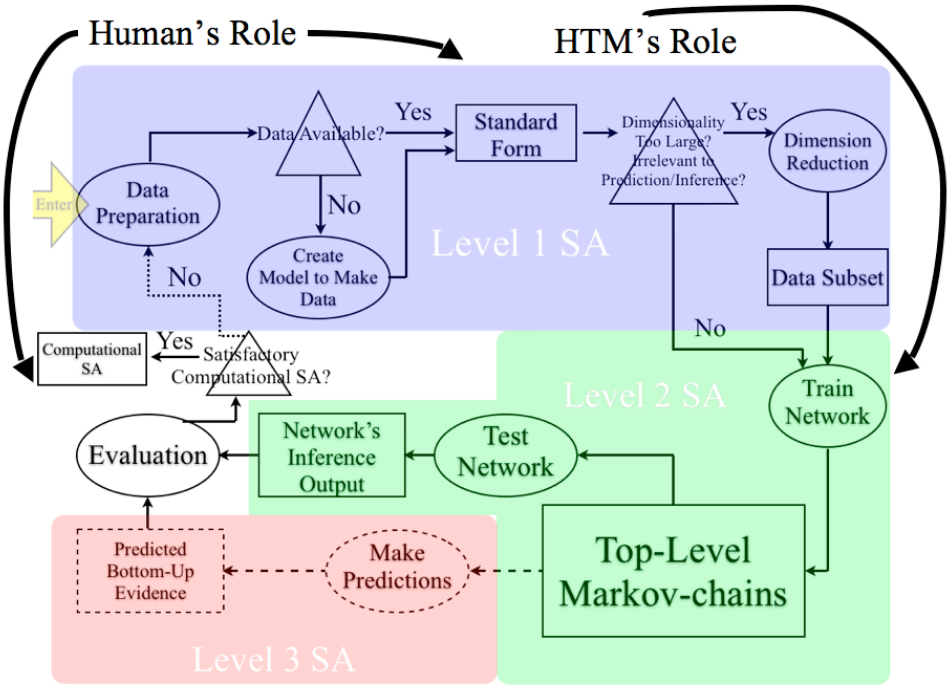


Figure 55: Modified Information Flow for HTM-based SA

The preceding experiments have focused exclusively on Level 2 SA. We have evaluated these results, asking all along whether the computational SA created by an HTM network was satisfactory. For some networks, this answer was 'no' and so the process repeated again through the beginning of Level 1 SA. We would prepare data and feed it into another network formed by evolutionary design from past implementations. The output to this information flow has been three computational SA implementations. These are the one unsupervised and the two supervised networks mentioned at the end of the previous section. These implementations have incorporated a human-performed Level 1 SA and an HTM-performed Level 2 SA.

There has been no analysis thus far of Level 3 SA. While this is certainly worthy research to follow, the goal in demonstrating these experiments for Level 2 SA has been to lay the foundation for application in other contexts. Specifically, we want to apply these methods to the Iraq context. Furthermore, since Level 3 SA follows and builds on Level 2 SA, the work done here would necessarily lead to Level 3 capabilities. But instead of probing ways to do this prediction, we now turn our attention to recognition in another context, the Iraq context. This context will not be as simple as the shockwaves context for many reasons. So we must carefully build on the analysis done here in the shockwaves context and recognize where our assumptions cannot be extended in what is to follow.

The Motivating Example: DoD SA in the Iraq context

We now proceed to the motivating scenario from the literature survey and our research questions. Specifically, we wish to create a computational situational awareness (SA) usable by the Department of Defense to gauge Iraq's stability during 2003-2008. This SA applies to the Iraq War from 2003-2008, which henceforth will be called the 'Iraq context'. In doing so, we will attempt to extend the work done thus far on SA in other contexts, such as the River (Waves) and shockwaves ones. At the same time, our methods must change to take into account the different nature of this problem. Specifically, this SA differs from the previous ones because validation is much trickier. While a goal (stability) and a context (Iraq 2003-2008) can be specified here as with previous scenarios, there is a fundamental difference. Specifically, it is the way to verify the accuracy of the computational SA formed in this context. To do so, a new method will be introduced that has influence from system dynamics and it is called *extreme-case bounding*. This method has assumptions built-in to it that creates fictitious extreme cases of stability, either extremely unstable or extremely stable. With these fictitious bounds, some insight into the actual progression of events in Iraq during 2003-2008 can be

obtained. Needless to say, this method is not perfect. Nevertheless, it provides an intriguing foothold in an avenue of computational SA that has thus far been difficult to probe concretely.

As with the previous context, we will follow the information flow (Figure 55) we have established for tackling computational SA. In the course of completing Level 1 SA, we will see that this is executed differently now because we actually have data on the context we wish to understand. Nevertheless, as mentioned with extreme-case bounding, some modeling will be needed to generate additional data. As before, all of the data preparation mechanisms will be put in place before training and testing HTM networks. There will be only minimal iteration between data generation and network training/testing. We will train and test HTM networks on these datasets – both extreme-case and actual – to make conclusions about the resulting Level 2 SA on the Iraq context. As we have mentioned, the evaluation will be a little more difficult, but the focus on extreme-case bounding will help in this regard. Then there will be a brief comparative analysis to actual reports by the DoD. What we will see though is that extreme-case bounding is a more substantive approach than such comparative analysis.

Preliminary Considerations: Coarse Graining, Description Length and Ergodicity

We saw in the literature review that temporal data on the Iraq context exists and this data determines our coarse graining. Some examples of this data were values of Iraqi civilian fatalities and the nationwide unemployment rate. As opposed to the shockwaves context, we cannot readily extract information on all of the relevant variables about the Iraq context. So, in forming a schema about the Iraq context, we are necessarily bound by the metrics we follow to track its behavior. In other words, our coarse graining – and consequently, our schema – is determined via the observables.

If we think back to the shockwaves context though, there was a similar case there as well. For instance, once the data was generated, our unsupervised schema (i.e., the

unsupervised $N = 4$ log-transformed Level 2 SA implementation) was built directly from observations on P, ρ, h and u. When there was supervision, this was another data source for learning. Both the $N = 2$ and $N = 4$ log-transformed Level 2 SA implementations were built from this combined data. To simulate data on flow we had to select a coarse graining. This choice restricted our observations on the flow to four properties (P, ρ, h and u). But these four properties completely described that context. So its description was relatively short.

This coarse graining not only simplified the shockwaves context but it also allowed it to satisfy the ergodic theorem. Recall that ergodicity means the system in question tends towards an end state. By confining our model to equilibrium flow, we made each time step the end state because that is what defines equilibrium. By creating data on this potentially complex system (supersonic bodies passing through real air) we made the problem abide by the ergodic theorem. We then generated data that satisfied this constraint and used it for training/inference. This is how we made SA of the shockwaves context.

Now, though, the situation is different. For instance, is there an ultimate end state to Iraq? How many metrics allow us to completely describe stability in Iraq? These two questions get to the real heart of why the Iraq context is a tough problem to analyze – why it is a *complex* problem. Considering the first question, we cannot pick a coarse graining of this context to box it into the ergodic theorem. Rather, we must extract what information we can from the data that is available. But as we saw with earlier unsupervised networks (e.g., for $N = 4$ log-transformed data), this information might not always be correct. With one particular network (schema) we had 97% accuracy in light of the goal to recognize flow types. How will we do now with a dataset that describes Iraq during 2003-2008 at a coarse graining for which there is no possibility of ergodic theorem application? We might need some additional way to evaluate our Level 2 SA formed from this data.

Considering the second question now, no one knows how many metrics are enough to describe Iraq's stability during 2003-2008. But the coarse graining that decides which variables are tracked should be such that certain general aspects of the issue are considered. Of course, this statement reflects somewhat of a top-down approach to gathering data. Yet, by specifying what aspects of the context are of importance to our primary goal (stability), we can select data that reflects a set of measurements towards this desired end state. For instance, the DoD focuses on political, economic and security aspects to stability. Consequently, we focus on data that pinpoints these aspects. Since such data exists, it will be used to describe these various aspects to stability. This is certainly not a complete description of the Iraq context. However, it is a quantitative description, and we shall try to use it in light of what we know from machine learning.

Finally, consideration of these two questions brings us to the conclusion that a semi-ergodic approach to stability analysis must ultimately be done here. This is, of course, the aforementioned extreme-case bounding. Why is this approach 'semi-ergodic'? It is because we create artificial situations that drive towards stable equilibrium or unstable equilibrium. For example, if the nationwide unemployment were to hit 100% then this would be the unstable equilibrium of this metric. Similar arguments apply to the other metrics, thereby allowing us to create equilibrium end-states artificially. By either training or evaluating networks' inference performance on such data, we are attempting to use HTM to quantitatively characterize the middle ground between stability and instability. Specifically, this is done via the probability distribution over Markov-chains. How well or how poorly this approach might work remains to be seen.

Level 1 SA: Data Preparation

As opposed to the River (Waves) and the shockwaves contexts, we actually have data on our current context of interest. But this is somewhat of a blessing in disguise because considerably more effort is needed to prepare the data into standard form. There

are four issues we must confront in doing so. First, the primary source ([72]-[83][84]) from which the data is extracted contains many blanks in the data, depending on how many metrics are used. So a set of metrics must be selected from the actual data that exhibits a minimal number of blanks. Second, the primary source has not prepared the data in a temporally structured format suitable for HTM learning. Recall, George writes, "if there is no temporal structure in the data, application of an HTM to that data need not give any generalization advantage." So the data must be arranged in this fashion. Specifically, observations at specific time intervals should follow one another. Third, the relative magnitudes of the chosen metrics will be necessary to consider. We saw this problem earlier in the shockwaves context, so we have some familiarity with it. But what worked there might not work here. Fourth and finally, one of the metrics we use to describe the Iraq context is only known within given bounds at each time step. Consequently, we must select a technique to get only one value at each time step, rather than a range. So we cautiously probe each of these steps with more detail now to prepare the data in standard form.

Minimal Blanks and How to Represent Them If Unavoidable

Recalling George's guidance for HTM generalization, we know that a training dataset must have temporal structure to it. Consequently, this means that it should have the fewest number of blanks possible. However, in the defense community, it is a common occurrence to have gaps in intelligence data. The available data on the Iraq context is no exception. It is therefore necessary to prepare the training data so that there is a minimal quantity of blanks. This requires picking metrics that are nearly continuous from 2003-2008. As a result, we have at least one constraining effect on what data we pick to describe the Iraq context.

Additionally, we will still have metrics for which there is a small but noticeable amount of blanks. So we need a way to differentiate these values from actual

observations of value equal to zero. In other words, we need to differentiate an observation of absence from an absence of observation, since a zero represents both in a dataset. In fact, this issue has been central to natural philosophy since the beginnings of the Scientific Revolution. George Berkeley notably commented on the famous quandary about a tree falling in the forest as follows [120]:

> The objects of sense exist only when they are perceived; the trees therefore are in the [forest] ... no longer than while there is somebody by to perceive them.

Similarly, an HTM network only trains and inferences on data coming into its receptive field, i.e., its perception of a context. But an inherent feature to the data contained in the Brookings report, and many other intelligence data sources, is that there are gaps in the information at unpredictable times for which interpolation could prove incorrect. This is a classic occlusion problem and it is rampant in data analysis [121]-[123]. We try to minimize its impact here though by picking a convention and following it.

We can see the problem directly by looking at two examples from the Brookings data compilation. Many metrics in the Brookings report are reported for every month from May 2003 to April 2008. But, consider our example from the literature review: the number of Iraqi civilian fatalities. This metric actually shows missing data in April 2008. We can see the potential problem directly by looking at the data from September 2006 to April 2008 (Table 9).

Table 9: Monthly Data from Brookings Report Showing Lack of Observations

Month	Iraqi_Civilian_Fatalities
Sep-06	3345
Oct-06	3709
Nov-06	3462
Dec-06	2914
Jan-07	3500
Feb-07	2700
Mar-07	2400
Apr-07	2500
May-07	2600
Jun-07	1950
Jul-07	2350
Aug-07	2000
Sep-07	1100
Oct-07	950
Nov-07	750
Dec-07	750
Jan-08	600
Feb-08	700
Mar-08	750
Apr-08	No Observation

So the number of Iraqi civilian fatalities would be zero in April 2008, unless we have a way to indicate numerically that there was no observation here. Consequently, what is needed is a way to preprocess the data such that the HTM does not learn an inaccurate coincidence pattern from such an instance.

Before a solution is posed to this minor issue, let us consider another related example from the Brookings data. This is a case where actual observations of a metric have the value of zero. We call these observations of absence. The number of rocket-propelled grenade (RPG) deaths for U.S. troops exhibits this. We can see it from looking at the data from April 2007 to April 2008 in Table 10.

Table 10: Monthly Data from Brookings Report Showing Observations of Absence

Month	Rocket_Propelled_Grenade_US_Troop_Deaths
Apr-07	1
May-07	0
Jun-07	4
Jul-07	2
Aug-07	4
Sep-07	3
Oct-07	0
Nov-07	0
Dec-07	0
Jan-08	1
Feb-08	1
Mar-08	1
Apr-08	2

As opposed to Table 9, Table 10 shows observations of no U.S. troop deaths from RPGs in May 2007 and October through December 2007. Thus, considering *in tandem* the examples of both Table 9 and Table 10, there is an issue when training and performing inference on data in which there are both absences of observations (e.g., Table 9) and observations of absence (e.g., Table 10). Consequently, we must differentiate these two cases somehow because both cases are treated the same way by default.

The proposed solution for dealing with such an issue is to make observations of absence equal to a small, yet finite, value, while absences of observation will remain having the value of zero. For instance, Table 10 would be transformed into Table 11, in which every zero has been replaced with *0.01*.

Table 11: Example of Replacement of Observations of Absence with Finite Values

Month	Rocket_Propelled_Grenade_US_Troop_Deaths
Apr-07	1
May-07	0.01
Jun-07	4
Jul-07	2
Aug-07	4
Sep-07	3
Oct-07	0.01
Nov-07	0.01
Dec-07	0.01
Jan-08	1
Feb-08	1
Mar-08	1
Apr-08	2

Thus, following the established solution for dealing with the difference between observing absence and lacking observations, the red section of Table 9 would be replaced with a *0*. In general, all instances such as those illustrated by Table 9 and Table 10 are handled in this way. Specifically, values of *0.01* replace values of a particular metric when *0* has been observed. A value of *0* is inserted for a particular metric when there has

been no observation on that metric's value during a particular time step. Furthermore, if the data is somehow transformed then this distinction scales accordingly.

Temporal Structure of Data

For HTM learning, we need temporally structured data. As mentioned briefly above, the time period over which there is a temporally consistent amount of data on the Iraq context is May 2003 to April 2008. This is at a monthly sampling rate. Since we want to attempt to train from real data, this temporal consistency is necessary. So *in tandem* with the earlier need for minimal blanks, this means that sixteen metrics can be chosen from the Brookings dataset [84], which is largely a compendium of DoD data [72]-[83]. These metrics are shown below (Table 12).

Table 12: Sixteen Metrics to Describe Iraq Context, 2003-2008

Metric	Units	Metric #
Iraqi_Civilian_Fatalities	persons	1
Multiple_Fatality_Bombings	persons	2
US_Troop_Fatalities	persons	3
Improvised_Explosive_Device_US_Troop_Deaths	persons	4
Car_Bomb_US_Troop_Deaths	persons	5
Mortar_and_Rocket_US_Troop_Deaths	persons	6
Rocket_Propelled_Grenade_US_Troop_Deaths	persons	7
Helicopter_Loss_US_Troop_Deaths	persons	8
Other_Hostile_Fire_US_Troop_Deaths	persons	9
Total_US_Troop_Deaths	persons	10
Attacks_on_Iraqi_Infrastructure_and_Personnel	incidents	11
Coalition_Troop_Strength	persons	12
Crude_Oil_Production	millions of barrels per day	13
Crude_Oil_Export	millions of barrels per day	14
Nationwide_Electricity	Megawatts	15
Nationwide_Unemployment_Rate	%	16

The table also shows an identifying number next to each metric. These numbers will be used in later discussions.

Relative Magnitudes of Data

Looking at the sixteen metrics chosen from the Brookings set, we see substantial differences in magnitude between some metrics. For instance, Coalition troop strength is ~10^5 and U.S. troop deaths from RPGs is ~1. Since these values do not experience sharp changes like we saw with the shockwaves context data, we can use infinity normalization here, instead of logarithm-transformation. Of course, we can also do a logarithm transformation as we did with the shockwaves context. But the infinity normalization will be the primary choice for preprocessing the data. So for this type of normalization, each metric is normalized by its maximum value over May 2003 to April 2008. We can see a profile of the changes in these values simultaneously now (Figure 56).

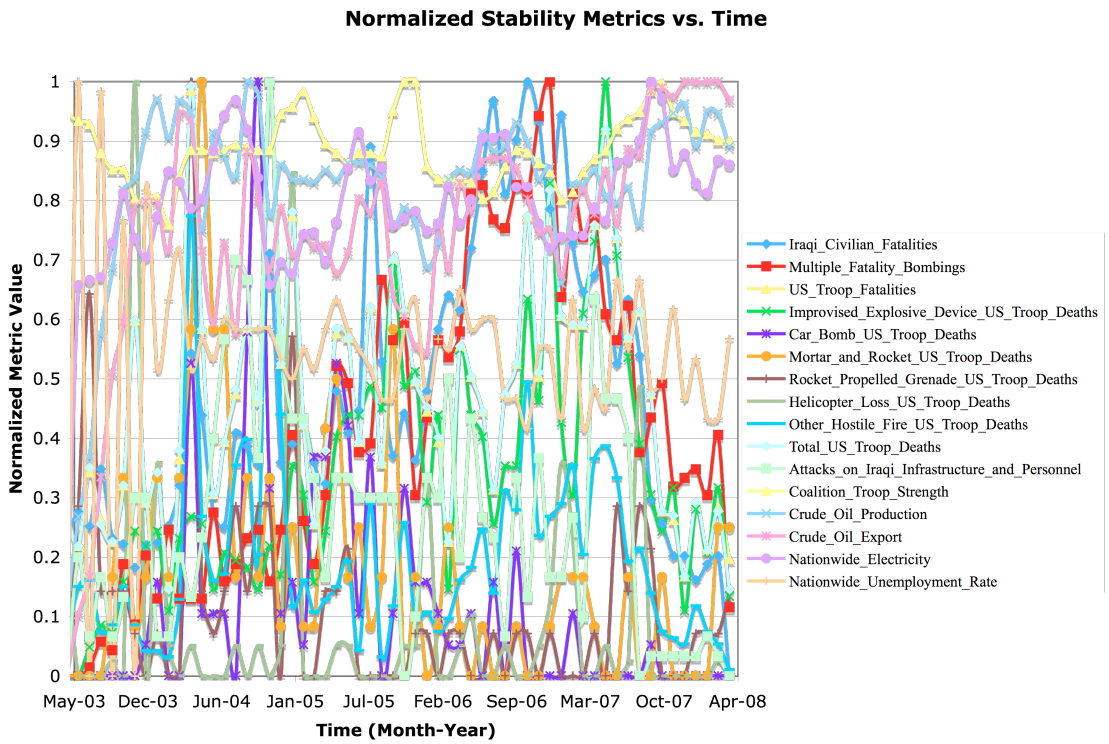


Figure 56: Normalized Stability Metrics - Time Evolution, 2003-2008

As the figure makes clear, there is no easy way to look at this data all together to determine trends in stability. Some trends oscillate substantially, while others do not. This is why we attempt to analyze this data with HTM.

Ranges of Values

The final issue to encounter with the data is when ranges of values are given. For instance, this is the case for the Iraqi nationwide unemployment rate. Consequently, we have assumed that the actual value lies somewhere between the given bounds. So a random value between the boundaries given by the Brookings report is chosen.

Summary of Data Preparation

Once these four issues are addressed, the data is ready to be put in standard form. It is combined in an Excel file and then cut-and-pasted into a space-separated text file, as we did with the shockwaves data. This gives us a $V_{properties}$ of $N = 16$ that describes the Iraq context. So we could proceed immediately to train an HTM and examine its inference. But having no other dataset on the Iraq context, this would not give tremendous insight to the Level 2 SA formed. Rather, we can remain within Level 1 SA of the information flow a little bit longer. While here, we will take the other path from the Boolean asking about whether we have data on the context we wish to study (see Figure 55). This other path tells us we must create a model to generate data on the Iraq context. So now we generate the extreme-cases mentioned earlier. Of course, for inference (and perhaps training), we want to make data that is of the same dimension as our current $V_{properties}$. So this model will consider only the $N = 16$ metrics shown above (Table 12).

More Level 1 SA: System Dynamics and Extreme-Case Bounding

Now that the actual data has been prepared in a format that is suitable for HTM generalization, we face a unique issue. As noted above, we do not know exactly how to validate the HTM output as easily as we had with previous contexts and goals. There is

no longer a visual basis for validation as there was with invariant visual pattern recognition. There are no longer results from equilibrium gas dynamics to validate the recognition of shocked flow. In short, there is no simple way to determine a stable situation from an unstable one, either by natural human information processing or a closed set of consistent equations. Consequently, our methods for analyzing HTM output will be different now from what they have been for previous contexts and goals. A possible method that is proposed and done here is called *extreme-case bounding*. This method uses artificial data that represents Iraq's stability becoming increasingly worse or better. It is then speculated that reality exists between these bounds. Consequently, we should be able to get a measure of how close or far a given month is to or from stability/instability. A one-dimensional representation of this concept is shown in Figure 57.

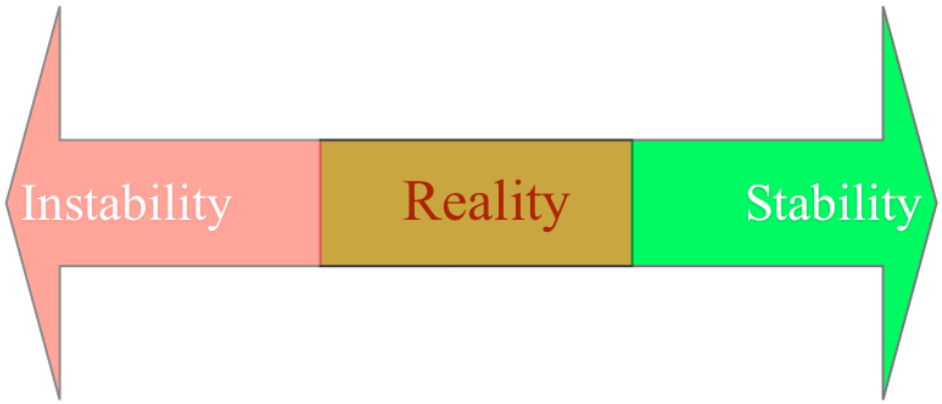


Figure 57: One-Dimensional Extreme-Case Bounding

But in the context of Iraq, the dimensionality will be higher than what Figure 57 shows. Instead, there will be extreme-cases for each of the sixteen dimensions by which stability is measured. To analyze this claim some tests are done and results are shown in subsequent sections. But first, it is important to specify the system dynamics model that will generate this extreme-case data.

So let us begin by describing how to make extreme-case data for this context. Just as with the shockwaves context, there are observables whose values obey certain relationships and factors that drive changes in their values. Recall within the shockwaves context that there were four observables that obeyed the conservation and state equations (Equation 1). Now, within the context of Iraq during 2003-2008, there are sixteen observables collected above into standard form. Also, in the shockwaves context, the presence of a supersonic body was a factor that drove changes in the metrics' values. But in the Iraq context these factors are slightly more complex. This is because in the shockwaves context a compressible fluid always shocks when the flow speed exceeds sonic conditions. So a supersonic body will *always* cause a shock, except in very special cases. But, in the Iraq context a factor will *not always* have the intended effect on one of the sixteen metrics.

To consider the issue of factors in more detail for the Iraq context, it is necessary to specify which factors are of interest. Of course, all factors are important, but we must recall whose SA is being modeled and analyzed here: the SA of the DoD. Just as with human operators of complex systems in Endsley's research, the entity forming and maintaining SA is constrained by the fact that the only factors it can use to change values of observables are those within its control. For instance, the pilot controls the plane with control surfaces, engine throttle, etc., even though there are other factors (e.g., turbulence) that can also affect observables crucial to his/her SA. Similarly, for DoD SA, only those factors within the DoD's control can be explicitly considered. Other factors can also be considered, but their effects are necessarily more approximate. Fortunately, each of the metrics that describe Iraq's stability has monotonic utility, as depicted in Figure 57. In other words, there are either desirable or undesirable changes to their values so the only question is which factors will have a dominating effect. If we assume that values change by some mechanism in time then an approach like system dynamics seems useful. So this is where we now turn.

Implementing Extreme-Cases: Lessons from System Dynamics

There are some guidelines that can be followed as we create these extreme-cases. Here is where the *Road Maps* series by Jay Forrester can be of tremendous help [124]. *Road Maps* is basically a how-to guide on system dynamics. In it, fundamental principles drive the discussion on how to build system dynamics models. But as we shall see, there are some points from this guide that differ from our goal with extreme-case bounding. Specifically, we are not as concerned with the interactions between observables as we are with driving them to extreme states. These differences are highlighted as we discuss each principle of interest.

The first point of distinction comes from the way feedback happens. Forrester writes, "System Principle #1 [is that the] feedback loop is the basic structural element of systems. ... Feedback loops are the building blocks of systems that are linked together to build more complex systems." Now it is important to notice the subtlety when applying this principle to the Iraq context. Recall that for the shockwaves context, there is negligible feedback between the supersonic body and the observables. This is because of the assumed gross difference in mass between a supersonic body and the gas through which it is passing. So there is no noticeable feedback loop between the factor (the body) and the observables (the gas properties). However, in building a model that creates an asymptotically stable or unstable condition for Iraq, there is feedback between the factors and the observables to consider. These effects are illustrated in Figure 58.

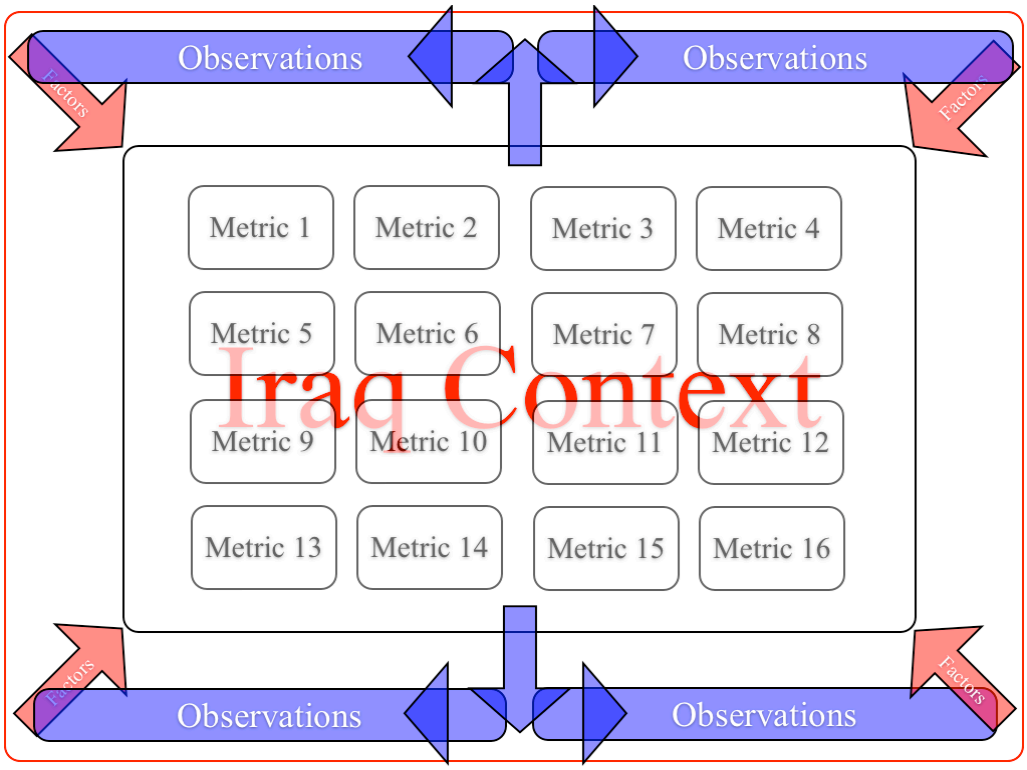


Figure 58: Schematic of Factors Influencing Observables

As the figure shows, observations of metrics instigate factors to influence their behavior in subsequent time steps. But it is not clear how observations motivate decisions on what factors should influence the system's behavior. This is in fact the whole point to analyzing NCW SA in this context. So by creating asymptotic cases for the Iraq context, it is assumed that the transfer function between observations and factors is such that Iraq drives towards or away from stability. There is no other assumption built into how feedback works in this system dynamics implementation.

To build a model that does this, levels and rates are fundamental things to consider for system substructure. Forrester writes, "System Principle #2 [is that levels] and rates are fundamental to loop substructure [and] to system structure. [This principle] is important to understand because all system dynamics models use levels and rates as loop substructure [.]" For instance, in the shockwaves context, the levels were the values of the observables. The rates manifested the factors by which the values changed. So the

supersonic body was a factor. Its rates were manifested through the conservation/state equations (Equation 1). For the Iraq context, a similar case exists. The levels are the values of the metrics. But there are no simple rate equations between the observables as there was for equilibrium air. Rather, the only rates that can be considered now are those coming from the factors driving each observable towards or away from stability. Necessarily, this perspective on rates incorporates the goal of DoD SA, i.e., making Iraq stable. Due to our assumed goal, there are two types of rates of interest to the DoD, those that increase stability and those that degrade it.

Given the focus on levels, rates and feedback, there are issues related to these worth noting. Specifically, the levels are reflections of rates working over time. Forrester writes, "System Principle #4 [is that levels] are accumulations (integrations). ... Levels accumulate the results of rates (actions) in the system." This point ties in with System Principle #13, which states, "Solution interval DT is in all level equations and no others. The DT ... is the time period in which the level is changed by the rate." So necessarily, levels accumulate rates' effects over a collection of DT intervals. This leads to System Principles #5 and #11, which state respectively, "Levels are changed only by rates" and "Levels completely describe the system condition. ... This completely describes the state of [the] system." So necessarily, the levels reflect the rates' work over time and at a given time describe the state of the system. If we recall Gell-Mann and other complex systems researchers, the goal there was to describe the system. The principles of system dynamics tell us that the levels (in our case, the observables) describe the overall system. Recall, this was the case for the shockwaves context, in which the observables together provide a complete description of the system. For the Iraq context, however, the idea is similar but the mechanics that lead to this description are fictitious. Consequently, the system that the levels describe is also fictitious, but it is desired that it be so. In particular, it is desired to be extremely fictitious for the purpose of bounding and so it is believed that

these simplifications are not problematic for describing Iraq's stability. Only analysis based on this approach will tell for sure.

There are some other insights to pull from system dynamics literature as well. For instance, Forrester writes, "System Principle #7 [is that the rates] depend only on levels and constants." Recall in the shockwaves context, the rates manifest factors' influence. But these factors can be broken down into those directly affected via levels (e.g., via conservation and state equations) and those affected via other mechanisms. These other mechanisms are called *constants* in system dynamics literature. For instance, an example of a constant in the shockwaves context was the presence of a supersonic body every x time steps. This value of x need not be fixed, i.e., constant. Only the affect of the body must be, i.e., it must cause a shock. But in our implementation to generate extreme cases in the Iraq context, it is not known how the levels affect the rates. For example, an increase in Iraqi civilian deaths (a level) could lead to more Coalition troops exerting effects (rates) in subsequent months on other stability metrics. Or it could not. The only constant factor is that there will be influence towards or away from stability via the observation-based feedback loop of Figure 58. It does not matter whether these effects come from Coalition troops, foreign insurgents, the media, etc. So the levels somehow affect the factors, which in turn affect the rates, but these dynamics are not known. So this model adheres to System Principle #7, but this connection is subtler in the Iraq context from what it was in the shockwaves context.

Some final points to consider from the system dynamics literature include caveats of first-order loops, conservation considerations, system boundaries and model validity. The goal of making this system dynamics model is to generate extreme cases of stability and instability. So the more important consideration is *that* it is done, rather than *how* it is done. For this purpose, first-order loops are sufficient. But Forrester warns in System Principle #10, "First-order loops exhibit exponential behavior." This means that it becomes increasingly likely across successive intervals DT for the levels to get out of

control. This was not a problem within the shockwaves context because the supersonic body was assumed to pass through space just once before the air reacted. Furthermore, the air reacted by reaching a new equilibrium condition, so the danger of exponential behavior was not as apparent. Of course, the air became successively hotter and more pressurized with each passing body, but the simulation ended before the properties of air violated our assumptions. Similarly, a model of the Iraq context is also susceptible to exponential behavior and so limits are imposed on the behavior as it evolves in time. In particular, it is assumed that most metrics do not exceed their peeks observed during 2003-2008. We say 'most' rather than 'all' because it might be desirable for some metrics to exceed their values during 2003-2008. For example, it would be desirable for Iraq's economic stability that crude oil production exceeds its values during 2003-2008. This is just how profit-driven economics work. If values of these metrics exceed their 2003-2008 peak by an order of magnitude then limits are placed on them. For instance, running two different kinds of extreme-case stability models can result in one implementation's results being bounded, while the other one's is not. But conversely, values of Iraqi civilian fatalities that exceed the population of the country are nonsensical. So its peak during 2003-2008 is assumed to provide a bound on its value during the simulation.

Table 13: Stability Metrics Bounded by Maxima/Minima/Percentage for Extreme-Case Instability

Metric	Units
Iraqi_Civilian_Fatalities	persons
Multiple_Fatality_Bombings	persons
Improvised_Explosive_Device_US_Troop_Deaths	persons
Car_Bomb_US_Troop_Deaths	persons
Mortar_and_Rocket_US_Troop_Deaths	persons
Rocket_Propelled_Grenade_US_Troop_Deaths	persons
Helicopter_Loss_US_Troop_Deaths	persons
Other_Hostile_Fire_US_Troop_Deaths	persons
Attacks_on_Iraqi_Infrastructure_and_Personnel	incidents
Coalition_Troop_Strength	persons
Nationwide_Unemployment_Rate	%

Table 14: Stability Metrics Bounded by Maxima/Minima/Percentage for Extreme-Case Stability

Metric	Units
Iraqi_Civilian_Fatalities	persons
Multiple_Fatality_Bombings	persons
Improvised_Explosive_Device_US_Troop_Deaths	persons
Car_Bomb_US_Troop_Deaths	persons
Mortar_and_Rocket_US_Troop_Deaths	persons
Rocket_Propelled_Grenade_US_Troop_Deaths	persons
Helicopter_Loss_US_Troop_Deaths	persons
Other_Hostile_Fire_US_Troop_Deaths	persons
Attacks_on_Iraqi_Infrastructure_and_Personnel	incidents
Coalition_Troop_Strength	persons
Crude_Oil_Production	millions of barrels per day
Crude_Oil_Export	millions of barrels per day
Nationwide_Electricity	Megawatts
Nationwide_Unemployment_Rate	%

A complete list of those metrics whose values are bounded in both extreme-case system dynamics models is shown in Table 13 and Table 14. The metrics highlighted with green are those metrics that are bounded in some models and not in others, due to the different exponential behaviors in each.

An issue that ties in with bounding exponential behavior is conservation. Forrester writes, “System Principle #6 [is that levels] exist in conservative subsystems. A conserved quantity has the property that it is never created or destroyed (within its system); it is only moved around.” For instance, the conservation and state equations kept the shockwaves observables in check so that enthalpy never got out of control (conservation of energy) or pressure never exceeded physical reality (conservation of momentum). Similarly, for extreme-case scenarios, a system dynamics model has conservation constraints between values of certain metrics. For instance, in one month, the number of U.S. troops killed in a helicopter loss cannot exceed the total number of U.S. troops killed. Similarly, the number of U.S. troops killed by an IED, other hostile fire, etc., cannot combine to form a number larger than the total number of U.S. troops killed. While such matters seem elementary, Forrester warns about exponential behavior from first-order loops, so the calculations purely from rates must be modified to account

for this and other conservation rules. A list of the conservation laws pertinent to the metrics of the Iraq context is shown below (Table 15).

Table 15: Conservation Laws Between Stability Metrics for All Extreme-Case Models

Conservation Law	
US_Troop_Fatalities = Total_US_Troop_Deaths	
Total_US_Troop_Deaths >=	Improvised_Explosive_Device_US_Troop_Deaths +
	Car_Bomb_US_Troop_Deaths +
	Mortar_and_Rocket_US_Troop_Deaths +
	Rocket_Propelled_Grenade_US_Troop_Deaths +
	Helicopter_Loss_US_Troop_Deaths +
	Other_Hostile_Fire_US_Troop_Deaths
Crude_Oil_Production > Crude_Oil_Export	

As we see from the table, the total number of U.S. troop deaths must be greater than or equal to the sum of indicated ways troops have died. This allows us to account for other troop deaths, such as accidents, etc. Also, we assume that the production of crude oil bounds its export. Finally, we assume that U.S. troop fatalities and number of troop deaths does not differ, even though this had been seen sporadically in the original data.

The last points to consider are related to each other. Forrest writes, "System Principle #21 [is that every] system has a closed boundary. In creating a model of a real system, any interaction which is essential to the behavior mode being investigated must be included inside the system boundary." Recall that for the shockwaves context there was a closed boundary around the system, such that exterior sources of mass, energy and momentum were not introduced (except in one perturbation case). Predominantly, the only external influence to this system was the periodic presence of a supersonic body. This interaction was fundamental to the ensuing equilibrium shock calculations. Similarly, the Iraq context has a closed boundary, though it is not obvious what this boundary is. Is it the country borders? Does it have to be a physical boundary? When thinking of boundaries to a system such as the Iraq context, these are possibilities. But it is essential to pick a boundary that includes any interaction essential to behavior.

Wherever it is for the Iraq context, we choose a boundary such that it encompasses any interactions that can degrade or improve Iraq's stability. Since extreme stability and instability are the behavior modes in question, it makes sense that all interactions leading to these extremes be considered. Of course, this does not force us to specify what these interactions are, but we must acknowledge their possible influence.

Having a closed boundary, the system can be modeled to investigate the behavior mode in question, i.e., extreme-cases of stability/instability. But how valid is the model? As Forrester writes, model validity is a relative matter. Specifically, he writes, "System Principle #26 [is that no] model is a perfect representation of a real object. A model is successful if it opens the road to improving the accuracy with which we can represent reality." So while a system boundary can be defined to consider relevant interactions, the accuracy of the resulting model is necessarily determined by such a choice. For instance, equilibrium flow is not the way real air behaves in the presence of a supersonic body. But it provides a model that improves the accuracy with which reality can be represented in a substantial set of cases. Similarly, in defining a system dynamics model of the Iraq context, it is not the aim to make a perfect representation of Iraq's conditions during 2003-2008. Rather, the aim is to create extreme-case boundaries that help training/inference of an HTM network. The aim is that a decision-maker can then use the network's output to better recognize actual stability conditions. As human factors tells us, better recognition – or categorization – of the situation leads to better decision-making. Necessarily, it remains to be seen to what extent such a model would open the road to improving the recognition of stability.

Implementing Extreme-Cases: Creating a System Dynamics Model

Having gone through the guidelines for creating an extreme-case system dynamics model, it now remains to implement them. Along the way, certain details as they apply to the Iraq context have been mentioned. It is useful now to describe some

actual models in their entirety. Such a description must cover everything from the feedback loops and rates to the time intervals and conserved quantities. This is done with an Excel macro interacting with a worksheet. For reference purposes, a copy of an example Excel macro used to generate data in one case is included in an appendix (C.1). This extreme-case model simulates *progressively* stable/unstable states in Iraq. Let us see now how this is done in light of the guidance from Forrester.

Starting from System Principle #1, the feedback loop between factors and metrics must be specified. Specifically, due to the monotonic utility of each metric, factors are either allowed to decrease or to increase each metric's value. These factors are split into two categories in this model: those factors performed to augment stability and those not. Given the specified goal of stability, it is assumed here that the U.S. military does all actions to promote stability, while actions done to degrade stability (including those inadvertent actions by the U.S. military) are considered separately. Of course, other agents and organizations promote stability in various forms. It is not just the U.S. military doing this. But we focus here on DoD SA and it is closely tied to the U.S. military's actions. These are actions in the control of the DoD and so the goal-oriented nature of SA requires us to focus on these actions. To implement the SA goals, it is assumed here that the Joint Urban Operations Joint Integrating Concept (JUO-JIC) provides an effective breakdown of the ways the U.S. military can attempt to augment stability in a troubled region like Iraq [125]. Of the twelve capabilities of the JUO-JIC, eight of them are directly pertinent to the metrics of the Iraq context. Two of these capabilities (9 & 10) have been combined into one, making a total of eleven capabilities available to affect change in the sixteen metrics. The capabilities as well as the ways they affect the corresponding metrics are shown below (Table 16).

Table 16: JUO-JIC Capabilities on Battlefield Assumed for System Dynamics Models

	Metric #															
Steps After Collect & Assess	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Integrate All Elements of Urban Operations																
Adapt Operations to Situation				-	-											
Maneuver to/through Area								+								
Apply Destructive Force to Hostile Elements	-	-	-	-	-	-	-	-	-	-	-	+	+	+	+	
Persuade Municipal Authorities to Cooperate																
Secure and Protect Urban Areas from Hostilities	-	-	-	-	-	-	-	-	-	-	-	+	+	+	+	
Isolate Portions of Urban System to Limit Collateral Influence	-												+	+	+	-
Affect Infrastructural Strengthening/Improvements												+	+	+	+	-
Facilitate Humanitarian Aid	-											+				-

This table specifies the interactions between metrics and those factors that promote stability. We see from this table that every metric is affected via a combination of these stability factors. The macro (C.1) then reads through this table and implements these factors each loop iteration. What is not shown in the table is that each of these metrics is also susceptible to factors that decrease stability. So we can take the opposites of the entrants in this table to see how decreased stability would manifest itself in these metrics' values. This information is directly coded into the macro and so it need not be read from an external table. In this way, factors either affect each metric in ways that either drives it towards stability or instability.

Following on the heels of feedback loop specifications, it necessary to specify the rates at which these loops operate. Recalling that our primary goal in creating this model is to simulate progressively extreme cases of stability and instability, the choice of rates is only important in so far as it should accomplish this goal. With this constraint in mind, it is assumed for this model that each stabilizing factor exerts a 5% change and that each destabilizing factor exerts a 20% change. To control which factors have a more dominating influence, these changes are subjected to probabilistic logic gates (see C.1). The primary variables then that determine whether the model is driven towards stability or instability are in these logic gates. For both types of factors, the probability of either stabilization or destabilization is set as an input via these two variables (see C.1).

Additionally, two logic gates rather than one are available by which destabilization can affect the metrics. This is done to approximate the potential for the U.S. military's inadvertent actions in their attempts to improve stability. These probabilistic logic gates computationally enact the rates of stabilization and destabilization.

It should be remembered that this is not proposed as a suitable model to explain Iraq's stability during 2003-2008. Rather, the focus of this model is to achieve progressively extreme cases of stability and instability. So these choices of rates and logic gates are only a means to this end. In fact, later we will create a simplified version of this model that ignores the logic gates and drives the states monotonically towards instability/stability via rates.

Looking back to the other System Principles related to levels, rates and feedback, we see that the model directly enacts them. For instance, System Principle #4 is satisfied via the update equations on each metric within the logic gates. System Principle #13 is satisfied because all changes to the levels happen across an interval DT, where DT equals one month, i.e., the same interval by which the actual data changes. System Principles #5 is directly satisfied because rates are the only effects on levels. System Principles #11 is satisfied because the levels at each time step are assumed to describe the system condition. The adherence of the model to System Principle #7 has already been discussed in detail above. Both the effects and the ways to curtail the first-order loops' exponential behavior (System Principle #10) have already been discussed above (Table 13 and Table 14). It should be noted that this accounting is not done via the macro, rather it is done as a post-processing step on the macro-generated data in Excel. The conservation laws of the Iraq context's subsystems (System Principle #6) were also discussed earlier (Table 15). As with the post-processing of exponential behavior, the conservation laws are also enacted in post-processing. Finally, System Principles #21 and #26 have been discussed in detail above.

So by implementing the *Road Maps*' framework for making a system dynamics model, it is possible to generate extreme cases for progressively less stable and progressively more stable versions of the Iraq context. We assume an initial condition for the system that is nearly what the state was in April 2008. We then propagate forward in time with the model. A sample of generated data is shown in an appendix (C.2). For the purpose of training/testing networks, one stable and one unstable extreme-case was run. The values for those variables that determine the probabilistic logic gates are shown in the example macro in an appendix (C.1). Now that progressively extreme-cases have been created, it is possible to return to HTM implementation details.

Level 2 SA: Training and Testing

With the ability to generate data for both progressively stable and unstable situations, as well as the actual time series of data on the Iraq context, it is possible to attempt HTM as an *unsupervised* machine learning mechanism. The aim now is to fuse the data and to extract possibly implicit meaning from it. We emphasize the unsupervised nature of the learning here because our goal is to extract implicit meaning and not to impose our possibly biased judgments. We recall briefly that the experiments in the shockwaves context have provided the glue between the rudimentary River (Waves) context and this one. In particular, those experiments showed us how well certain HTM networks could train and infer from N properties describing behavior at one point in space. Now, we attempt to extend this approach to a system that is not ergodic, not separable into components and not completely observable. So we employ now a two-pronged approach to the training and analysis of these HTM networks. The first is to use actual data and the second is to use progressively extreme-case data. But a question arises as to how the data should be used. Should the HTM be trained with the real data and tested on the extreme-case data? Or, should it be trained with the extreme-cases and be

tested on the actual data? Without clear guidance, both methods are attempted and evaluated here.

The evaluation of this computational SA is less straightforward than previous examples. Consequently, some additional techniques are employed here to probe the SA formed about the Iraq context. For instance, we do not know if too many or too few metrics are being used here to describe the Iraq context. So we will see what the effects on training/inference are when the number of metrics is reduced from $N = 16$. Also, we will examine the degree to which information is hierarchically stored in intermediate levels of the networks. Finally, we will consider alternative ways of feeding the data into the networks to see what effects – if any – there are on the simulated SA.

While one of these techniques has been employed in previous contexts, the others have not thus far been of use. For instance, we saw the effects of a reduced number of metrics (from $N = 4$ to $N = 2$) in the shockwaves context networks. But conversely, we have not probed the hierarchical storage of information in-depth so far. Why? Recall that our purpose in using HTM for schema formation has been its declared ability to condense information into hierarchies of both space and time. We have only hinted at this occurring in the shockwaves context's networks. But in that analysis we had focused primarily on the top-level node. There was not as great of a need to probe intermediate levels' performance because we knew what to expect at the top-level, since this node covered the whole receptive field. Consequently, the top-level node implemented our goal (e.g., flow pattern recognition or shock recognition) and it was up to our evaluation to determine if it was done correctly. However, this does not mean that we ignored the analysis of intermediate nodes. For instance, we used bottom-level nodes' feed-forward inference as a basis for describing higher-level nodes' inference output during the perturbation test cases. It was just not overtly described earlier. Now, for the Iraq context, we will test hierarchical storage more directly. We do this now because it is not clear what the top-level node's output *should* be, as it was for the shockwaves, Waves or

Pictures networks. One possible outcome of this analysis is that it might in turn help us to identify what aspects of the Iraq context are not well observed. This would then provide the beginnings of a feedback mechanism with Level 1 SA to search for more data. We will implement and analyze one possible feedback mechanism between Levels 1 & 2 SA. In particular, we will see that there is some room for improvement in the extreme-case data used to train a network. Consequently, we will return briefly to Level 1 SA after having done Level 2. Necessarily, this is not the only possible feedback mechanism, but as we will see, it does help strengthen the credibility of the Level 2 SA formed here computationally.

Finally, there is one other tool at our disposal that has not been mentioned. Specifically, we can use the DoD reports on stability in Iraq [72]-[83] in a comparative analysis. This comparison would provide somewhat of a mapping between inference on the actual data and the descriptions given in these reports. As we will see, the use of extreme-case bounding provides more concrete analysis than such comparisons. Without further preliminaries, we now turn to our experiments and their analysis.

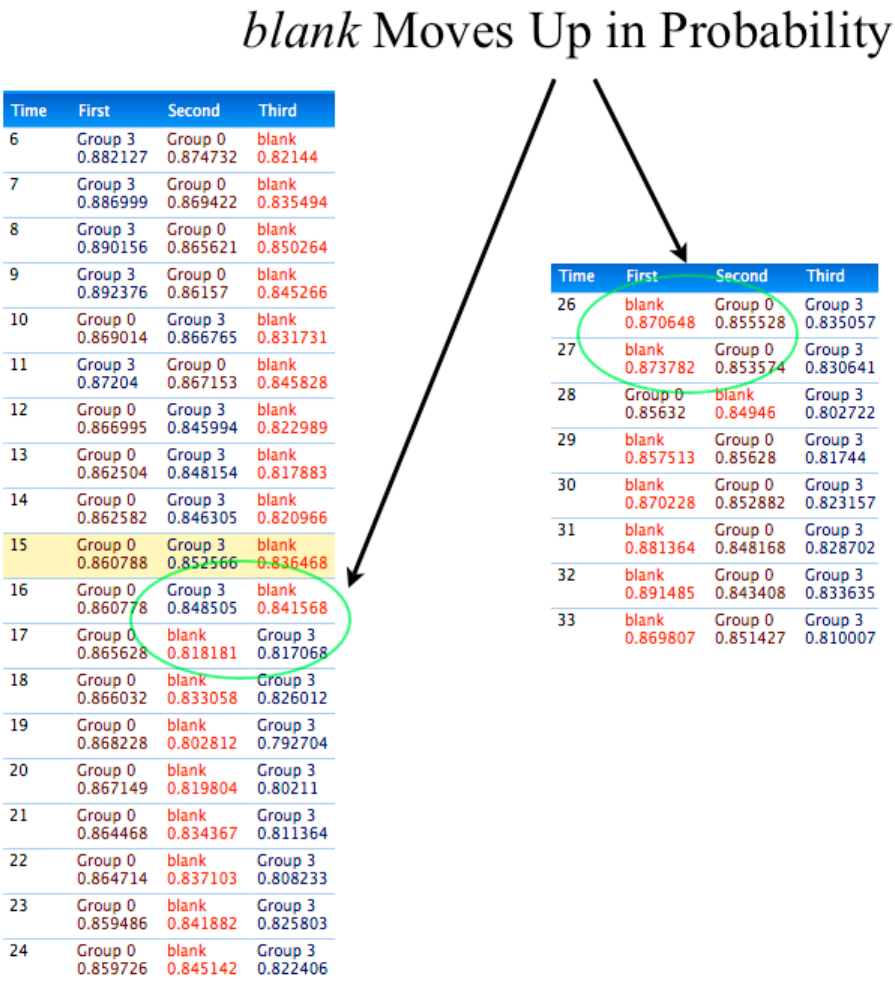
Experiment #1: Train on Actual Data and Test with Extreme-Cases

First, we attempt to create Level 2 SA directly from the actual data on the Iraq context. This means that we attempt to train an HTM network on an $N = 16\ V_{properties}$ vector space made of the metrics' infinity-normalized values from May 2003 to April 2008 (C.3). These metrics have already been discussed (Table 12) and we assume for now that they suitably describe the Iraq context. Later, we will test how reducing the number of metrics changes the formed schema. Using an evolutionary approach to HTM network design, we begin from networks similar to ones we have used thus far. We start with an unsupervised network that is nearly identical to the one used in the shockwaves context. The full network parameters are available in an appendix (C.4). To summarize, it is a three-level (4 x 2 x 1) network and the bottom-level value of *maxDistance* has been

reduced since the shockwaves implementation. If we train this network on this data then the top-level node forms thirty-nine patterns ($C^{3,1}$) and eighteen Markov-chains ($G^{3,1}$). We begin from a working hypothesis that these top-level Markov-chains represent patterns in the observable Iraq context, as they did before in the shockwaves, Waves and Pictures contexts. As we proceed with inference analysis, we will be able to test this hypothesis. Along the way, we will look at the intermediate nodes to do so. Having eighteen presumed gradations of patterns in the Iraq context, we proceed to test this claim.

We now use our extreme-cases to test this trained network's abilities. We begin with the dataset on progressively increasing stability and we follow the top-level node's feed-forward inference output. In particular, we consider $\max\left[\lambda_t(g_r)\right]$ at each time point to indicate the most likely Markov-chain at each value of *t*. The complete history of $\lambda_t(g_r)$ is shown in an appendix (C.5) for the three most likely Markov-chains (g_r). From this inference history, and assuming that the data represents progressively increasing stability, we see that $\max\left[\lambda_t(g_r)\right]$ indicates *g0* until *t = 33*, at which point *g17* takes over. Looking at the complete history of $\lambda_t(g_r)$, we see that this Markov-chain consistently moves up in terms of likelihood for $t \in [25,32]$. So as stability increased (via the metrics' values), the network increasingly recognizes *g17* as a more likely pattern (i.e., Markov-chain) in the Iraq context. But then at *t = 37* and $t \in [43,60]$, we see *g0* return, when in fact we designed the data to continue in the general trend of stability. Why does this happen?

Let us look to the intermediate levels' feed-forward inference for an explanation. First we examine the inference history of the left most node of the bottom level ($N^{1,1}$) for $t \in [6,33]$ (Figure 59).



Time	First	Second	Third
6	Group 3 0.882127	Group 0 0.874732	blank 0.82144
7	Group 3 0.886999	Group 0 0.869422	blank 0.835494
8	Group 3 0.890156	Group 0 0.865621	blank 0.850264
9	Group 3 0.892376	Group 0 0.86157	blank 0.845266
10	Group 0 0.869014	Group 3 0.866765	blank 0.831731
11	Group 3 0.87204	Group 0 0.867153	blank 0.845828
12	Group 0 0.866995	Group 3 0.845994	blank 0.822989
13	Group 0 0.862504	Group 3 0.848154	blank 0.817883
14	Group 0 0.862582	Group 3 0.846305	blank 0.820966
15	Group 0 0.860788	Group 3 0.852566	blank 0.836468
16	Group 0 0.860778	Group 3 0.848505	blank 0.841568
17	Group 0 0.865628	blank 0.818181	Group 3 0.817068
18	Group 0 0.866032	blank 0.833058	Group 3 0.826012
19	Group 0 0.868228	blank 0.802812	Group 3 0.792704
20	Group 0 0.867149	blank 0.819804	Group 3 0.80211
21	Group 0 0.864468	blank 0.834367	Group 3 0.811364
22	Group 0 0.864714	blank 0.837103	Group 3 0.808233
23	Group 0 0.859486	blank 0.841882	Group 3 0.825803
24	Group 0 0.859726	blank 0.845142	Group 3 0.822406

Time	First	Second	Third
26	blank 0.870648	Group 0 0.855528	Group 3 0.835057
27	blank 0.873782	Group 0 0.853574	Group 3 0.830641
28	Group 0 0.85632	blank 0.84946	Group 3 0.802722
29	blank 0.857513	Group 0 0.85628	Group 3 0.81744
30	blank 0.870228	Group 0 0.852882	Group 3 0.823157
31	blank 0.881364	Group 0 0.848168	Group 3 0.828702
32	blank 0.891485	Group 0 0.843408	Group 3 0.833635
33	blank 0.869807	Group 0 0.851427	Group 3 0.810007

Figure 59: Failing Recognition in Leftmost Bottom-Level Node

From this figure, we see that a Markov-chain called *blank* consistently moves up to be the most likely Markov-chain. What does this mean? This node learned only four Markov-chains from the data in its receptive field. This receptive field covers values of Iraqi civilian fatalities, multiple fatality bombings, U.S. troop fatalities and IED U.S. troop deaths. But as the stability increases forward in time for this dataset, the values for these metrics go further outside of the ranges that were seen from the actual data. For instance, at no time during 2003-2008 did the number of U.S. troop fatalities drop below *21* (March 2004). But the extreme-case for stability passes that value after the sixth month. As similar situations occur for the other metrics, the node becomes decreasingly able to

recognize the bottom-up evidence ($\bar{e}_t$). Consequently, we see a problem in the feed-forward inference at higher levels because of this lack of recognition at the bottom-level. For instance, if we look at the parent to this node, then we see for $t \in [6,25]$ that the *blank* Markov-chain is also increasingly likely. This parent node had learned twelve Markov-chains ($G^{2,1}$) and at $t = 24$ $\max\left[\lambda_t(g_r)\right]$ indicates none of these are most likely, given the bottom-up evidence coming from its children nodes. As this lack of recognition propagates up the network, we produce increasingly inconsistent results in the top-level node from $t = 25$ onward. Returning to the inference history of the top three Markov-chains of this node (C.5), we can see this. Why does this happen at $t = 25$ and not immediately at $t = 24$, when the bottom-level node could not recognize any of its Markov-chains? This is presumed to be because of how the feedback inference (π) refines the feed-forward inference. But as the feed-forward inference becomes increasingly ambiguous, there is nothing for the feedback inference to refine. So we can conclude from this observation that it is important to have training experience that provides boundaries on the situations about which we wish to infer. Our first attempt at training and testing a network has violated this conclusion. We have trained from data that blends evidence of both stability and instability without giving boundaries from which to learn.

So we already see inconsistency in the network's ability to recognize progressively stable states. Let us now confirm this insight by looking at its abilities to recognize progressively less stable states. The complete top-level node inference history on this dataset can be found in an appendix (C.6). We can see from $\max\left[\lambda_t(g_r)\right]$ over all values of t that the *blank* Markov-chain is indicated as early as $t = 15$. Now, the trouble comes from the right half of this node's receptive field ($N^{2,2}$) and its child nodes. At $t = 15$, this node's value of $\max\left[\lambda_t(g_r)\right]$ indicates a *blank* Markov-chain. Its children nodes indicate this too within 5-10 time steps of this value of t. For instance, if we look at the right-most bottom-level node then we see why. $N^{1,3}$ learned eight Markov-chains ($G^{1,3}$)

from data on crude oil production, crude oil export, nationwide electricity and nationwide unemployment rate. But at *t = 17* of the progressively unstable data, only crude oil production's value is within the bounds seen during 2003-2008 (see C.7). After two more time steps, this metric then goes outside of its bounds seen during 2003-2008. And consequently, at *t = 19*, $\max\left[\lambda_t(g_r)\right]$ indicates the *blank* Markov-chain, i.e., the node recognizes none of its learned Markov-chains. So as with inference on the progressively stable states, we see inconsistency in how the top-level node recognizes progressively unstable states. This seems to be because the data used for training does not cover a large enough breadth of both stable and unstable cases.

Experiment #2: Train on Extreme-Cases and Test with Actual Data

Building on these conclusions, let us try a different training strategy: we will train the network on progressively stable and unstable states of the Iraq context. Then we will observe its distribution over top-level Markov-chains to see what aspects of stability and instability are recognized in the bottom-up evidence at each point. This might then give us a probability-based assessment of how much *towards* stability or how much *towards* instability Iraq is going at any given *t* during 2003-2008. With this modification to our earlier working hypothesis, we shall try this strategy. We will retain the same network that we have used above, but will train and test it this way.

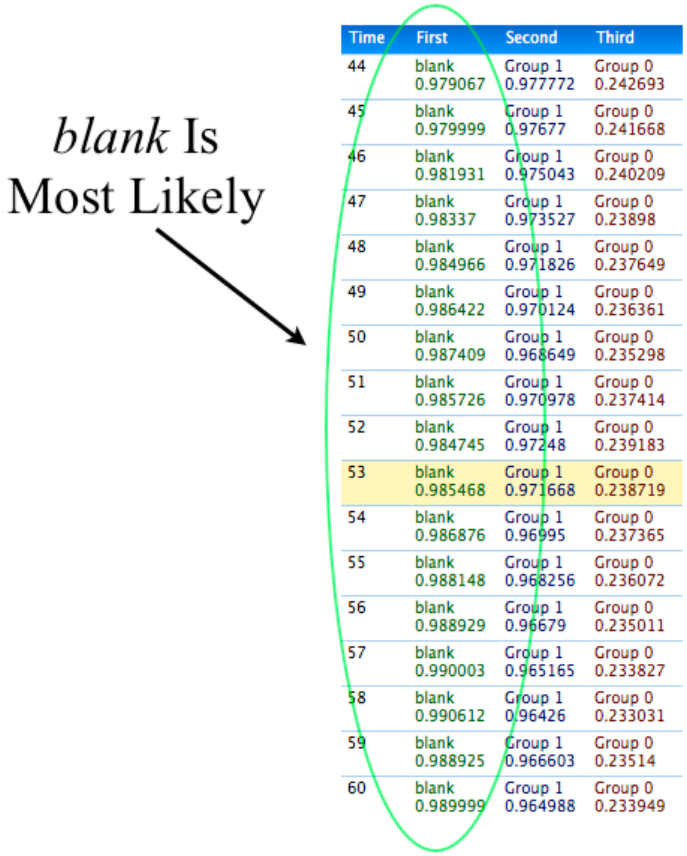
Since we train the network on two datasets, some modification to our prior data preparation and training approaches is needed. Specifically, we must concatenate these datasets with each other. The ordering of how this is done should not matter in terms of schema formation. We have chosen to start with progressively stable and then unstable states. Whichever order is chosen, it is necessary to indicate a break between these progressions. This is done so the network does not learn the break as a significant event among a sequence of them. Furthermore, the network must know not to incorporate this event into its schema formation. This is done with the *detectBlanks* parameter and so it is

activated here. We create a break in the data with a row of zeroes (i.e., blanks) and so this forms our progressively stable and then unstable dataset (see appendix C.8).

Now we train a network on this dataset and assess inference results. The network parameters can be found in an appendix (C.9), but they are the same as the ones used in the previous experiment. The top-level node learned sixteen coincidence patterns ($C^{3,1}$) and eight Markov-chains ($G^{3,1}$). Since we know that the training data indicates progressively stable and then (after the break) progressively unstable states of Iraq, we should be able to identify Markov-chains that indicate these states from inference on the training data. For instance, we look at $\max\left[\lambda_t(g_r)\right]$ for $t \in [0,60]$, i.e., as the states become progressively stable (see most likely Markov-chain in C.10). We see that after *g0*, the following Markov-chains are sequentially most likely in light of the bottom-up evidence: *g3, g5, g0, g3, g6, g2, g4* and *g2*. Then after the break, *g0* returns and then *g1* is indicated from $t = 111$ onward. This is somewhat of an unplanned result in recognition of the progressively stable states. For instance, if *g0* were a state from the early part of the progressively stable history then why would it reappear after *g3* and *g5*? Similarly, why would *g3* reappear after *g0*? This could indicate that the network's training with regards to more stable states is somewhat flawed. Conversely, we look at the top-node's recognition of progressively unstable states and see what we expected: the node recognizes *g0* and then *g1* as time moves forward. In other words, the top-level node unquestionably recognizes a progression towards instability but it does not recognize one as clearly for stability. So why does recognition of progressively unstable states seem better than that of progressively stable states?

We get the answer from examining the lower-level nodes of the network. Let us look at the bottom-level nodes during inference of this training data. First, we will look at the progressively stable states' inference, and then at the progressively unstable states' inference. If we look at the progressively stable states' inference in bottom-level nodes

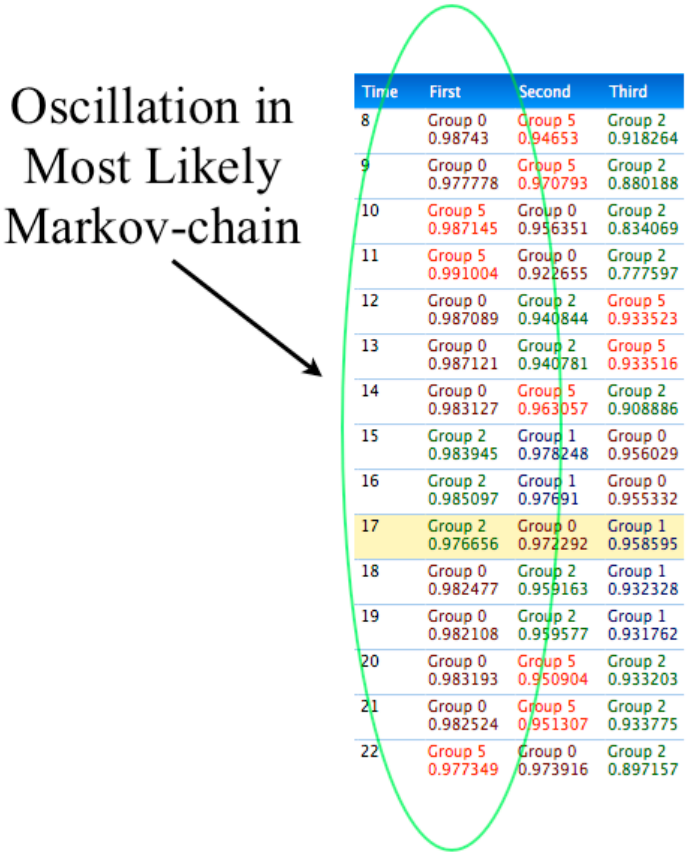
then we see from where the problem comes. Consider the bottom-level $N^{1,2}$ node's inference for $t \in [44,60]$ (Figure 60).



Time	First	Second	Third
44	blank 0.979067	Group 1 0.977772	Group 0 0.242693
45	blank 0.979999	Group 1 0.97677	Group 0 0.241668
46	blank 0.981931	Group 1 0.975043	Group 0 0.240209
47	blank 0.98337	Group 1 0.973527	Group 0 0.23898
48	blank 0.984966	Group 1 0.971826	Group 0 0.237649
49	blank 0.986422	Group 1 0.970124	Group 0 0.236361
50	blank 0.987409	Group 1 0.968649	Group 0 0.235298
51	blank 0.985726	Group 1 0.970978	Group 0 0.237414
52	blank 0.984745	Group 1 0.97248	Group 0 0.239183
53	blank 0.985468	Group 1 0.971668	Group 0 0.238719
54	blank 0.986876	Group 1 0.96995	Group 0 0.237365
55	blank 0.988148	Group 1 0.968256	Group 0 0.236072
56	blank 0.988929	Group 1 0.96679	Group 0 0.235011
57	blank 0.990003	Group 1 0.965165	Group 0 0.233827
58	blank 0.990612	Group 1 0.96426	Group 0 0.233031
59	blank 0.988925	Group 1 0.966603	Group 0 0.23514
60	blank 0.989999	Group 1 0.964988	Group 0 0.233949

Figure 60: Excerpt of Inference History for Second Bottom-Level Node from Left

We see that $\max\left[\lambda_t(g_r)\right]$ for $t \in [44,60]$ indicates a *blank*, i.e., neither of the two Markov-chains ($G^{2,1}$) learned by this node are recognized during this interval. As we know from earlier analysis, this inference propagates up the network, causing a problem at the top-level node. The second problem can be seen from the $N^{1,3}$ node's inference for $t \in [8,22]$ (Figure 61).



Time	First	Second	Third
8	Group 0 0.98743	Group 5 0.94653	Group 2 0.918264
9	Group 0 0.977778	Group 5 0.970793	Group 2 0.880188
10	Group 5 0.987145	Group 0 0.956351	Group 2 0.834069
11	Group 5 0.991004	Group 0 0.922655	Group 2 0.777597
12	Group 0 0.987089	Group 2 0.940844	Group 5 0.933523
13	Group 0 0.987121	Group 2 0.940781	Group 5 0.933516
14	Group 0 0.983127	Group 5 0.963057	Group 2 0.908886
15	Group 2 0.983945	Group 1 0.978248	Group 0 0.956029
16	Group 2 0.985097	Group 1 0.97691	Group 0 0.955332
17	Group 2 0.976656	Group 0 0.972292	Group 1 0.958595
18	Group 0 0.982477	Group 2 0.959163	Group 1 0.932328
19	Group 0 0.982108	Group 2 0.959577	Group 1 0.931762
20	Group 0 0.983193	Group 5 0.950904	Group 2 0.933203
21	Group 0 0.982524	Group 5 0.951307	Group 2 0.933775
22	Group 5 0.977349	Group 0 0.973916	Group 2 0.897157

Figure 61: Oscillation in Third Bottom-Level Node's Inference During Progressive Stability

For an alleged progressively stable situation, we see here that the Markov-chain indicated via $\max\left[\lambda_t(g_r)\right]$ oscillates. The node recognizes *g0, g5, g0, g2, g0,* etc. We see why this is so when we look at the way the bottom-up evidence ($\bar{e}_t$) changes over this time period (Figure 62).

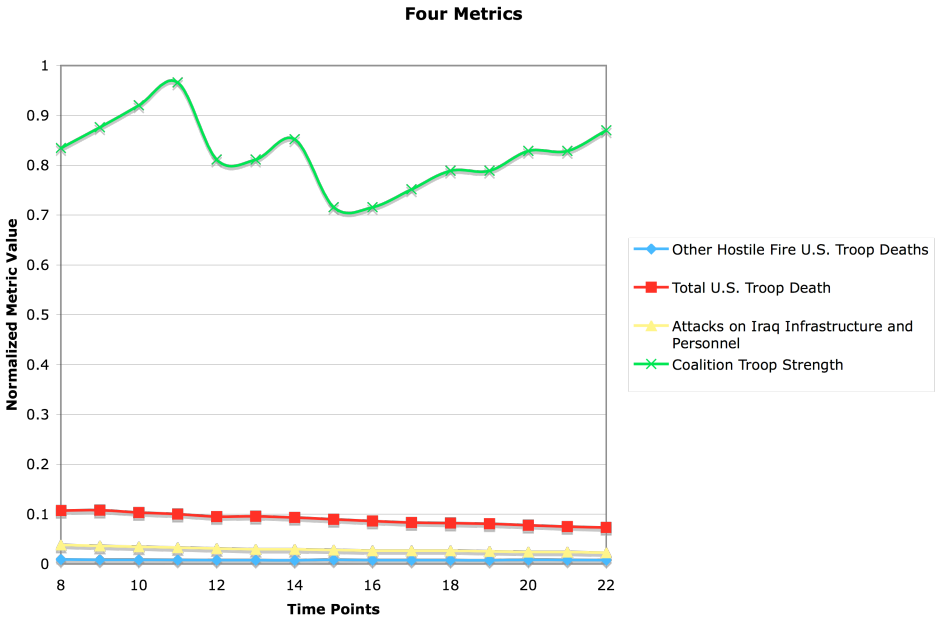


Figure 62: Oscillation in Time Series Witnessed by Third Bottom-Level Node

We see from this figure that the values of all four metrics oscillate. Furthermore, this was the data used for training. We would not have such trends if the data were monotonically approaching either stability or instability. We have to recall though that the system dynamics model did not progress to stability or instability this way. Rather, the probabilistic logic gates attempted to mimic the unpredictable nature of the Iraq context. Recall that each of the metrics has monotonic utility. For instance, from the DoD's perspective, stability is proportional to a decrease in total U.S. troop deaths. But we see from Figure 62 that the trend is not monotonic. Instead, there is oscillation in this metric, as there is with the other three – the oscillation in Coalition troop strength being most noticeable. With this non-monotonic bottom-up evidence ($\bar{e}_t$), we see why both the learning and subsequent feed-forward inference behave as they do. It is clear now why the node's feed-forward inference is confused over this time period. There is no clear monotonicity in $\bar{e}_t$. As a result, the feed-forward inference is also not monotonically indicative of progressive stability. For instance, the sharp drops in coalition troop strength

at *t = 12, 15* line up with the changes in $\max\left[\lambda_t(g_r)\right]$ for $N^{1,3}$. It is not known whether monotonic or progressively extreme-cases produce better SA of the Iraq context. Later, we will see how monotonically extreme-cases can affect learning and inference. Right now though, we will see if training on progressively unstable cases suffers from the same problem.

So now we look at the recognition of progressively unstable states. Specifically, we look at the bottom-level to see why inference of this data went as expected. Each bottom-level node shows monotonic indications from $\max\left[\lambda_t(g_r)\right]$. For instance, the $N^{1,3}$ node indicates *g2, g1* and then *g0*. There is no oscillation in the Markov-chains indicated by $\max\left[\lambda_t(g_r)\right]$. This is the case for the other three bottom-level nodes. As with the progressively stable states, we see that this occurs because of the bottom-up evidence coming into the nodes. Specifically, the bottom-up evidence is monotonic towards instability. We can see this from Figure 63.

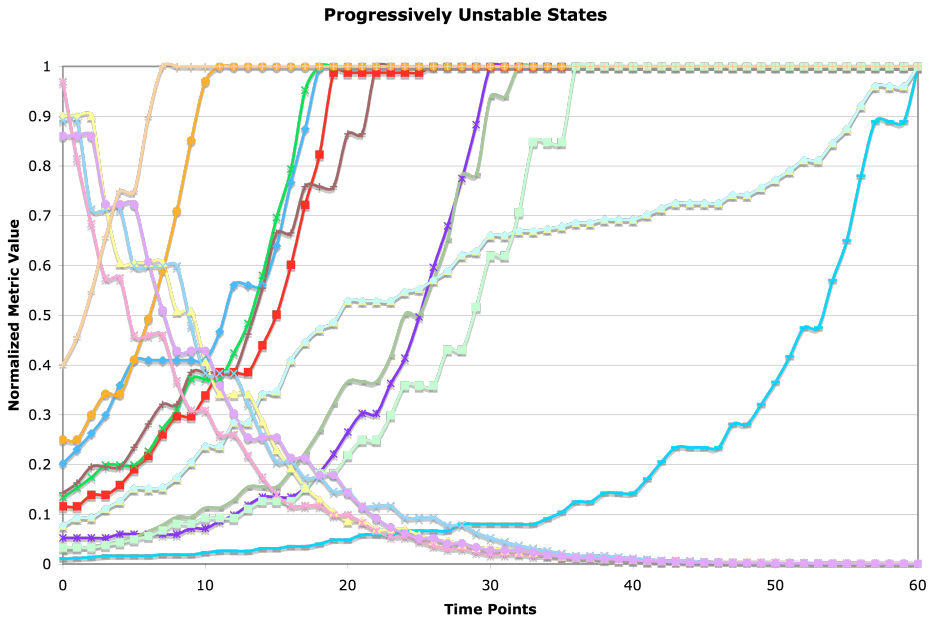


Figure 63: Time Series Witnessed by All Bottom-Level Nodes

From the figure, we see that each metric moves monotonically in time until reaching a bounding value. This happens in spite of the fact the SD model was implemented with the probabilistic logic gates discussed earlier. Furthermore, the feed-forward inference from the bottom-level nodes reflects this. This monotonic indication of Markov-chains feeds up to the second- and third-level, giving a monotonic indication at the top-level. Consequently, the top-level node indicates *g1* in the latter stages of inference on extreme-case instability training data. So that is why the network exhibits monotonic results about progressively unstable states, rather than what was seen with progressively stable ones. So we can justifiably conclude then that *g1* is a state that is closer to our extreme-case of instability than any other Markov-chain learned in the top level. In effect, *g1* is an "unstable" state.

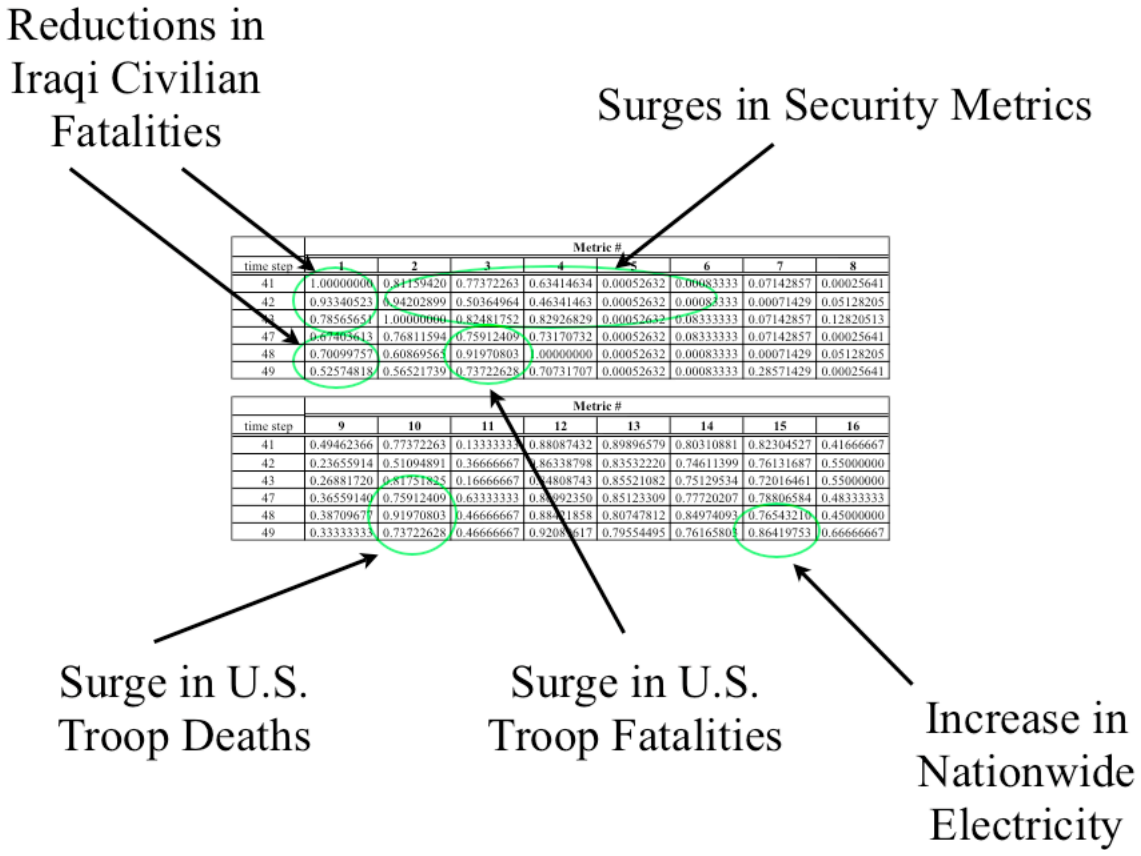
With this knowledge, let us see if this unstable state is recognized at any time point in the real data. Will the actual evidence concur with this assessment? We now look at the inference performance on the actual data in light of having trained a network on these extreme-cases. For instance, we can track the recognition of *g1* in this inference task. Recall, *g1* is the Markov-chain recognized as the Iraq context data drives towards instability. In fact, by tracking the probability of *g1* relative to other Markov-chains, we might even see how close or far each data point is from instability. It is important to note that we do not say that we might be able to recognize stability. This is due to the ambiguity in the progressively stable training data. But the scale between instability and stability is monotonic (as depicted in Figure 57), so recognition of stability is implied.

We proceed to analyze the network's inference of the real data. A complete inference history $\lambda_t(g_r)$ for $t \in [0,59]$ of the top-level can be found in an appendix (C.11). If we look at $\max\left[\lambda_t\left(g_r\right)\right]$ for these values of t then we see some intriguing findings about the Iraq context. We see that at $t = 18$, *g1* is the most likely Markov-chain, given the bottom-up evidence ($\bar{e}_{t=18}$). This is because, at this value of t, four of the sixteen metrics inversely proportional to stability hit a maximum. These metrics are U.S. troop fatalities,

other hostile fire U.S. troop deaths, total U.S. troop deaths, and attacks on Iraqi infrastructure and personnel. The maxima in these metrics occurred in November 2004 in Iraq. There is no corroboration of this from DoD reports because the bill that required the reports to be made to Congress did not go into effect for another six months [126]. In fact, it is quite possible that observations like this in November 2004 prompted more rigorous future assessments of the stability situation in Iraq. Whatever the case may be, this network learned what a progressively unstable situation in the Iraq context looks like from simulated evidence (i.e., the SD model). We then use this knowledge base to pass an assessment of actual data every month. For November 2004, the network recognized an unstable state with the highest probability. But what can it tell us about other times for the Iraq context?

Let us look at the movement of *g1* at other times to see how well it indicates movement away or towards instability. For instance, for $t \in \{[41,43], [47,49]\}$, *g1* is the second or third most likely Markov-chain in light of $\bar{e}_t$. These time periods stand out from other ones because *g1* has the lowest probability over the majority of the time points for the entire data set. But at these time intervals, *g1* jumps up to second or third most likely. This should indicate greater instability at these time points in comparison to the majority of the time points. However, does this assessment follow from the evidence? Consider the bottom-up evidence for $t \in \{[41,43], [47,49]\}$ (Table 17). We see for $t \in [41,43]$ that there are substantial surges in U.S. troop deaths and other security metrics. Conversely, the number of Iraqi civilian fatalities drops. For $t \in [47,49]$, the evidence is less clear.

Table 17: Ambiguous Bottom-up Evidence Categorized by Network



	Metric #							
time step	1	2	3	4	5	6	7	8
41	1.00000000	0.841159420	0.77372263	0.63414634	0.00052632	0.00083333	0.07142857	0.00025641
42	0.93340523	0.94202899	0.50364964	0.46341463	0.00052632	0.00083333	0.00071429	0.05128205
43	0.78565651	1.00000000	0.82481752	0.82926829	0.00052632	0.08333333	0.07142857	0.12820513
47	0.67403613	0.76811594	0.75912409	0.73170732	0.00052632	0.08333333	0.07142857	0.00025641
48	0.70099757	0.60869563	0.91970803	1.00000000	0.00052632	0.00083333	0.00071429	0.05128205
49	0.52574818	0.56521739	0.73722628	0.70731707	0.00052632	0.00083333	0.28571429	0.00025641

	Metric #							
time step	9	10	11	12	13	14	15	16
41	0.49462366	0.77372263	0.13333333	0.88087432	0.89896579	0.80310881	0.82304527	0.41666667
42	0.23655914	0.51094891	0.36666667	0.86338798	0.83532220	0.74611399	0.76131687	0.55000000
43	0.26881720	0.81751825	0.16666667	0.74808743	0.85521082	0.75129534	0.72016461	0.55000000
47	0.36559140	0.75912409	0.63333333	0.82292350	0.85123309	0.77720207	0.78806584	0.48333333
48	0.38709677	0.91970803	0.46666667	0.88421858	0.80747812	0.84974093	0.76543210	0.45000000
49	0.33333333	0.73722628	0.46666667	0.92087617	0.79554495	0.76165803	0.86419753	0.66666667

For instance, over these months, the number of Iraqi civilian fatalities drops and nationwide electricity increases, but the number of total U.S. troop deaths is high relative to other months. Despite these gradations in evidence, the HTM passes its judgment on instability with a probability that climbs during these time periods. So we see here a good demonstration of how the HTM could be used to resolve conflicting pieces of evidence into a solid assessment of instability.

As *g1* drops in probability, it becomes less clear to verify this indication of instability. For instance, for $t \in [44,46]$, are these time points less unstable because *g1* has dropped in probability from the other time steps? As with the other time points, there is bottom-up data both for and against instability. For instance, the number of U.S. troop fatalities drops during this time period and so does the number of Iraqi civilian fatalities. But there are monotonic trends in stability seen in crude oil production, crude oil export

and nationwide electricity. So it is possible that $t \in [44,46]$ is a less unstable period than $t \in \{[41,43], [47,49]\}$. Nonetheless, it is difficult to confirm this claim because of the bottom-up evidence ambiguity.

Alternative Network Settings

Let us see though how the results change if we tinker with the network parameters. For instance, we can decrease the value of *maxDistance* in the bottom-level node and see how this change propagates through the network. From training on the same data as the previous network, the top-level node learned nineteen coincidence patterns ($C^{3,1}$) and ten Markov-chains ($G^{3,1}$). This is a higher number of patterns and Markov-chains than what was seen before. It is largely attributable to the smaller threshold for storing distinct patterns at the bottom-level of the network. This network's bottom-level has *maxDistance = 0.008*, rather than *0.009*. A complete inference history of this new network on the training data can be seen in an appendix (C.12). To summarize, we see the same problem in progressively stable states. Specifically, the network learns from the data oscillation and so we do not get a clear picture of progressively stable states from the top-level Markov-chains. Looking at the progressively unstable states, we see a possibly confusing result at first. In particular, $\max\left[\lambda_t(g_r)\right]$ indicates the following progression of Markov-chains: *g1, g9, g0* and *g3*. Although this progression is monotonic, it is slightly confusing if we observe that *g0* starts the progressively stable states. How could a progressively stable state now be recognized as unstable? This is not a problem with the network. Rather, it is a problem borne from the training data. In particular, the data does not begin at the same initial condition for progressively unstable states that it did for progressively stable ones. This is an artifact of the infinity normalization because different normalizations are used in both progressively extreme-cases. For the most part this effect is minimal, and when it does occur we must simply acknowledge it so that our judgment of inference performance is not errantly led astray. This is one such case. So,

knowing this, we ignore the first three time points of the progressively unstable data (i.e., when *g1* and *g9* were most active). Consequently, this indicates *g3* as being close to instability in the same way that *g1* was for the previous network. Notice that the label has changed but the schema condensed into this label is the same. This schema concerns states of the Iraq context that are suitably close to instability. But will this recognized state exhibit similar performance to what was seen earlier for *g1* at certain time points of the real data?

Let us examine this network's inference of real data as we did for the previous one. Specifically, we will follow *g3* during the same time intervals we did earlier. We see from the inference history that the network still recognizes $t = 18$ (i.e., November 2004) as an unstable state. For $t \in \{[41,43], [47,49]\}$, we see similar results to what we saw with the previous network. There are some differences now though. For instance, *g3* is the fourth most likely Markov-chain at $t = 42$ and the seventh most likely at $t = 49$. Recall that *g1* (the "unstable" Markov-chain from the previous network) was the third most likely Markov-chain earlier. This is quite obviously an effect of there being a lower value of *maxDistance* in the bottom-level nodes. It is possible that the lower value of *maxDistance* is more selective about recognizing "unstable" states because of the lower threshold used in the bottom level when forming coincidence patterns. Whatever the reason, there is only a small effect from altering the network this way, and the gross features of it being able to recognize evidence of instability holds.

Further tests of altering the same bottom-level parameters reveal similar results. We reduce the *maxDistance* parameter again and see similar results to the previous two. This leads us to believe that these three networks have some ability to recognize states of the Iraq context that tend towards instability.

But how much do we really know about these networks? How general are these results and how would other implementations compare? For instance, we could use fewer metrics, or we could feed the data in differently. Are we building hierarchies of

information on politico-economic stability? – Of security? To answer these questions, we proceed to employ the techniques discussed earlier. Specifically, these techniques will examine if fewer metrics can be used to produce similar results; how results change if the data is fed in differently; and, to what extent information is stored hierarchically. We will begin with an analysis of the effects from reducing the number of metrics.

Reduced Number of Metrics

We now reduce the number of metrics used to train and test an HTM on the Iraq context. Effectively, this reduces the number of sensors used to monitor the context. As with the shockwaves context, we will see degradation in top-level inference performance for fewer metrics. Here though, a slower degradation can be seen because of the larger original number of metrics.

We begin by reducing the dimensionality of the $V_{properties}$ vector space from $N = 16$ to $N = 8$. The original $N = 16$ dataset was rich on data pertaining to various ways that U.S. troops were killed in combat, such as deaths related to RPGs, helicopters and other hostile fire. Now though, we wrap all of this data into the total number of U.S. troop deaths. Also, we no longer have redundant data on the number of U.S. troop deaths, as we had earlier with U.S. troop fatalities. We keep three out of four politico-economic metrics, eliminating only crude oil export, which is dependent on crude oil production. The final metrics for the $N = 8$ $V_{properties}$ vector space are shown below (Table 18).

Table 18: Reduced Number of Metrics (*N = 8*) to Describe Iraq Context

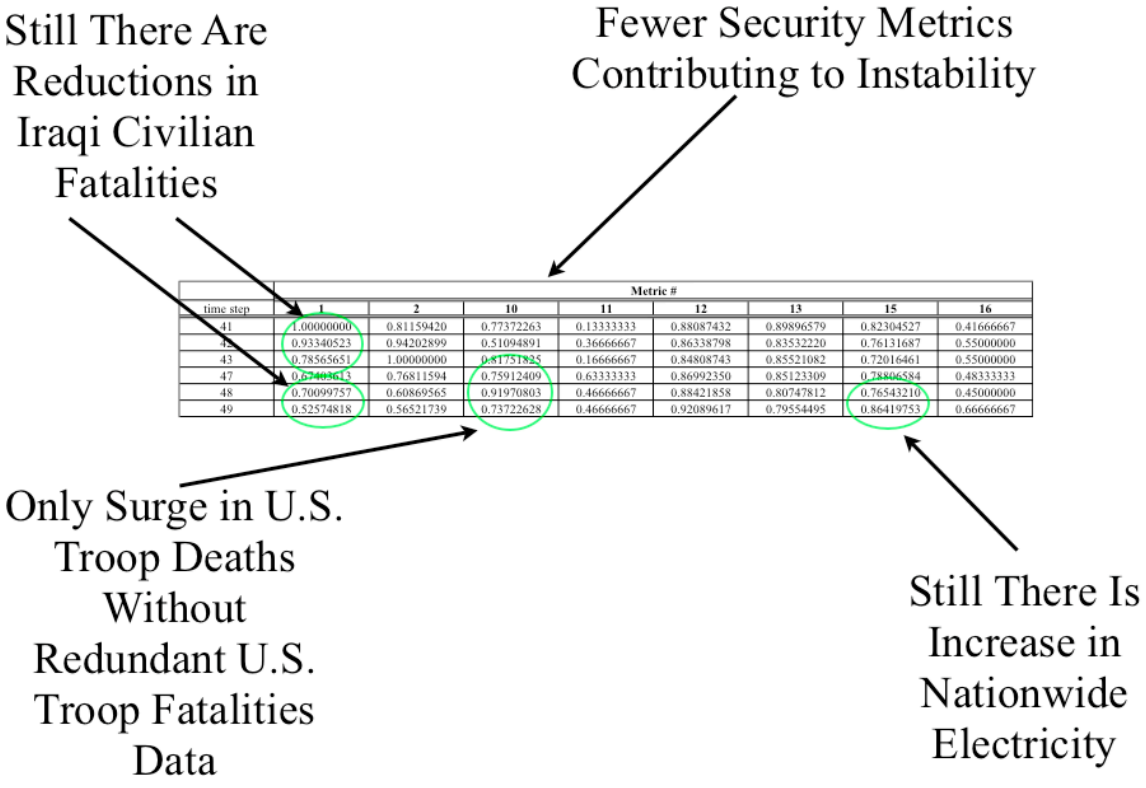
Metric	Units	Metric #
Iraqi_Civilian_Fatalities	persons	1
Multiple_Fatality_Bombings	persons	2
~~US_Troop_Fatalities~~	~~persons~~	~~3~~
~~Improvised_Explosive_Device_US_Troop_Deaths~~	~~persons~~	~~4~~
~~Car_Bomb_US_Troop_Deaths~~	~~persons~~	~~5~~
~~Mortar_and_Rocket_US_Troop_Deaths~~	~~persons~~	~~6~~
~~Rocket_Propelled_Grenade_US_Troop_Deaths~~	~~persons~~	~~7~~
~~Helicopter_Loss_US_Troop_Deaths~~	~~persons~~	~~8~~
~~Other_Hostile_Fire_US_Troop_Deaths~~	~~persons~~	~~9~~
Total_US_Troop_Deaths	persons	10
Attacks_on_Iraqi_Infrastructure_and_Personnel	incidents	11
Coalition_Troop_Strength	persons	12
Crude_Oil_Production	millions of barrels per day	13
~~Crude_Oil_Export~~	~~millions of barrels per day~~	~~14~~
Nationwide_Electricity	Megawatts	15
Nationwide_Unemployment_Rate	%	16

The ones that have been omitted are also indicated. All of the original metric numbers are kept for ease of continuity with previous work.

We train the network on progressively extreme-case data and then assess inference results. The complete network parameters can be found in an appendix (C.13), but the only difference is the number of metrics coming into each sensor. This network has learned fewer coincidence patterns and Markov-chains in the top-level node than its *N = 16* counterpart. It has learned just eleven coincidence patterns ($C^{3,1}$) and six Markov-chains ($G^{3,1}$). Analysis of the top-level inference on training data reveals the same oscillation in progressively stable states and the same monotonicity in the progressively unstable states. Consequently, *g1* is recognized to be an "unstable" state. When we perform inference with this network on real data (C.14), we see that at *t = 18 g1* is the most likely Markov-chain again. This is exactly what had been seen for the *N = 16* network. Recall, for the original *N = 16* network (C.11), *g1* was in the top three Markov-chains for $t \in \{[41,43], [47,49]\}$. Now, *g1* is the third most likely Markov-chain in light of $\bar{e}_t$ at $t \in \{42, [47,49]\}$. We see this slight change in the recognition of the "unstable"

state because of the fewer channels by which the evidence enters the network. We can see this by looking at the actual values of the metrics over these time points (Table 19).

Table 19: Less Bottom-up Evidence Indicating Instability for *N = 8*



	Metric #							
time step	1	2	10	11	12	13	15	16
41	1.00000000	0.81159420	0.77372263	0.13333333	0.88087432	0.89896579	0.82304527	0.41666667
42	0.93340523	0.94202899	0.51094891	0.36666667	0.86338798	0.83532220	0.76131687	0.55000000
43	0.78565651	1.00000000	0.81751825	0.16666667	0.84808743	0.85521082	0.72016461	0.55000000
47	0.67403613	0.76811594	0.75912409	0.63333333	0.86992350	0.85123309	0.78806584	0.48333333
48	0.70099757	0.60869565	0.91970803	0.46666667	0.88421858	0.80747812	0.76543210	0.45000000
49	0.52574818	0.56521739	0.73722628	0.46666667	0.92089617	0.79554495	0.86419753	0.66666667

For instance, there is not the redundant increase in U.S. troop deaths from the measurement of U.S. troop fatalities at $t = 41, 43$. Consequently, these time points are not recognized to be as "unstable" as $t = 42$. The bottom-up evidence is what causes this. We see for instance that attacks on Iraqi infrastructure and personnel experience a local maximum, and that crude oil production hits a local minimum across $t \in [41,43]$. With fewer metrics to confound these trends, the recognition of the bottom-up evidence through intermediate levels zooms in on this perceived instability at $t = 42$. So, despite slight difference, we see some substantial overlap between the $N = 16$ and the $N = 8$ networks in terms of their abilities to recognize states associated with instability.

If we reduce the metrics even more then the results become less conclusive. For instance, we reduce our number of metrics to $N = 4$ and assess results. We keep only Iraqi civilian fatalities, coalition troop strength, crude oil production, and nationwide electricity (see Table 20).

Table 20: Reduced Number of Metrics ($N = 4$) to Describe Iraq Context

Metric	Units	Metric #
Iraqi_Civilian_Fatalities	persons	1
~~Multiple_Fatality_Bombings~~	~~persons~~	~~2~~
~~US_Troop_Fatalities~~	~~persons~~	~~3~~
~~Improvised_Explosive_Device_US_Troop_Deaths~~	~~persons~~	~~4~~
~~Car_Bomb_US_Troop_Deaths~~	~~persons~~	~~5~~
~~Mortar_and_Rocket_US_Troop_Deaths~~	~~persons~~	~~6~~
~~Rocket_Propelled_Grenade_US_Troop_Deaths~~	~~persons~~	~~7~~
~~Helicopter_Loss_US_Troop_Deaths~~	~~persons~~	~~8~~
~~Other_Hostile_Fire_US_Troop_Deaths~~	~~persons~~	~~9~~
~~Total_US_Troop_Deaths~~	~~persons~~	~~10~~
~~Attacks_on_Iraqi_Infrastructure_and_Personnel~~	~~incidents~~	~~11~~
Coalition_Troop_Strength	persons	12
Crude_Oil_Production	millions of barrels per day	13
~~Crude_Oil_Export~~	~~millions of barrels per day~~	~~14~~
~~Nationwide_Electricity~~	~~Megawatts~~	~~15~~
~~Nationwide_Unemployment_Rate~~	~~%~~	~~16~~

Two networks were tested this way: one is the original network topology and the other is the same network with the second level removed. Neither network was able to give tractable results. Both networks suffered from the same problem concerning progressively stable states as previous ones. These networks though could not learn to recognize "unstable" states as the corresponding $N = 16$ and $N = 8$ ones could. This is presumably because, in the progressively unstable situation, the number of Iraqi civilian fatalities goes to a maximum, and all other metrics go to zero. Consequently, there is simply not enough non-zero data in the network's receptive field to perform inference.

So we see here a valuable lesson about evidence-based reasoning. There is certainly a lower bound beyond which it becomes increasingly inaccurate to ignore more metrics that describe a context. This lower bound exists somewhere between $N = 8$ and N

$= 4$ for the Iraq context. So we quantitatively also see the merit of considering more than a handful of metrics when assessing such a context. This is surely an important result to be considered in decision-making and we see some proof of it here.

Alternative Ways of Feeding Data

The next two analysis techniques are examined in similar ways. We do so by testing different ways of feeding data into the network. Specifically, we will see that alternative ways of feeding data into the network are explicitly tied to the hierarchical way by which the training data is condensed into a knowledge base. We see the effects during inference of both training and testing data.

So let us proceed now to describe the permutations we make to the data. Let us label each of the $N = 16$ metrics with a number as done earlier (Table 12). Let us look at two random permutations of these metrics to reorder the data sets (Table 21).

Table 21: First Two Permutations of Metric Order

	Permutation #	
	3	**4**
Metric #	12	3
	11	1
	4	15
	6	8
	1	13
	7	5
	13	6
	14	4
	15	14
	5	12
	16	16
	3	11
	9	10
	8	7
	10	9
	2	2

Here we will examine the consequences on learning and inference when the data is fed into a network these two ways. For the first permutation, the same network (C.9) that was used earlier learned more coincidence patterns and more Markov-chains. Now, the network's top-level node learned twenty-two coincidence patterns ($C^{3,1}$) and ten Markov-chains ($G^{3,1}$). So we already see that a random permutation of the metrics in the sensors alters the network's perception and consequent condensation of data.

Let us look at the inference of this network more closely. The complete inference history on the training data set is shown in an appendix (C.15). We see some interesting features here. First, the reordering does not help the monotonic learning of progressively stable states. For instance, during progressively stable data, we see the following groups indicated by $\max\left[\lambda_t(g_r)\right]$ in the top-level node: *g0, g4, g5, g4, g8, g9, g0, g6, g0, g1, g7*. So as before, we see a non-monotonic progression of Markov-chains during inference of progressively stable data. This lends further support to our earlier claim that such non-monotonicity is linked to the bottom-up evidence from which it learns, rather than the network itself. However, the progressively unstable states, still maintain monotonicity. For instance, $\max\left[\lambda_t(g_r)\right]$ indicates *g0, g2, g3*, indicating *g3* as a state suitably close to instability. How does the inference change when we look at actual data?

As before, we can use this trained network to perform inference on actual data, but we see slightly different results from earlier. The complete inference history can be found in an appendix (C.16). As before, at *t = 18*, *g3* is the third most likely Markov-chain, whereas for the original ordering it was the first most likely. Now, at *t = 11*, we see *g3* is the first most likely Markov-chain. When we look back to the original network's inference at this time point (C.11), we see only a slight jump in the probability of the "unstable" Markov-chain (*g1*). And if we look at inference for $t \in \{[41,43], [47,49]\}$ then we see nearly no indication of the instability of these states. There is only a slight increase in the probability of *g3* relative to other Markov-chains at *t = 43*. Otherwise, *g3*

remains the tenth most likely Markov-chain during this period. What we see here is interesting. The original ordering of the data yielded recognition of unstable states a certain way. Now this permutation of the data leads the network to recognize instability another way. These different abilities to recognize states obviously come from the different condensation of data that occurs in both networks. For instance, politico-economic metrics are hierarchically condensed with security metrics. Surge tracking metrics are hierarchically condensed with politico-economic metrics. The result is a hierarchical condensation of arbitrary groups of metrics. Nevertheless, there are some similarities in the states these networks do recognize. Let us look at one more permutation to see these effects in more detail.

We proceed then to a second random permutation of the data ordering. So we feed this permutation into the same network (see network parameters in C.9) that was used earlier and analyze results. This network's top-level node learned sixteen coincidence patterns ($C^{3,1}$) and seven Markov-chains ($G^{3,1}$). This only differs from the original network in there now being one fewer Markov-chain. During inference of progressively stable states, we see the same non-monotonicity in the most likely Markov-chains of this node. For inference during progressively unstable states, we see two familiar things. The first is that the first three time points should be ignored because they reflect the normalization artifact mentioned earlier. After these points, we have the same monotonicity we have seen earlier. The "unstable" state here is *g1*. When performing inference on the real data, *g1* is the most likely Markov-chain at $t = 11$ again. This is the same result we saw for the previous permutation. At $t = 18$, there is a jump in the probability of *g1* as there was for the first permutation. But *g1* is not the most likely Markov-chain as it had been with the original ordering of the metrics. Similar to the previous permutation, *g1* has the lowest probability for $t \in \{[41,43], [47,49]\}$.

To summarize these permutation tests, we see some interesting things. First, the ordering of the metrics into the sensors obviously matters for training a network. This is

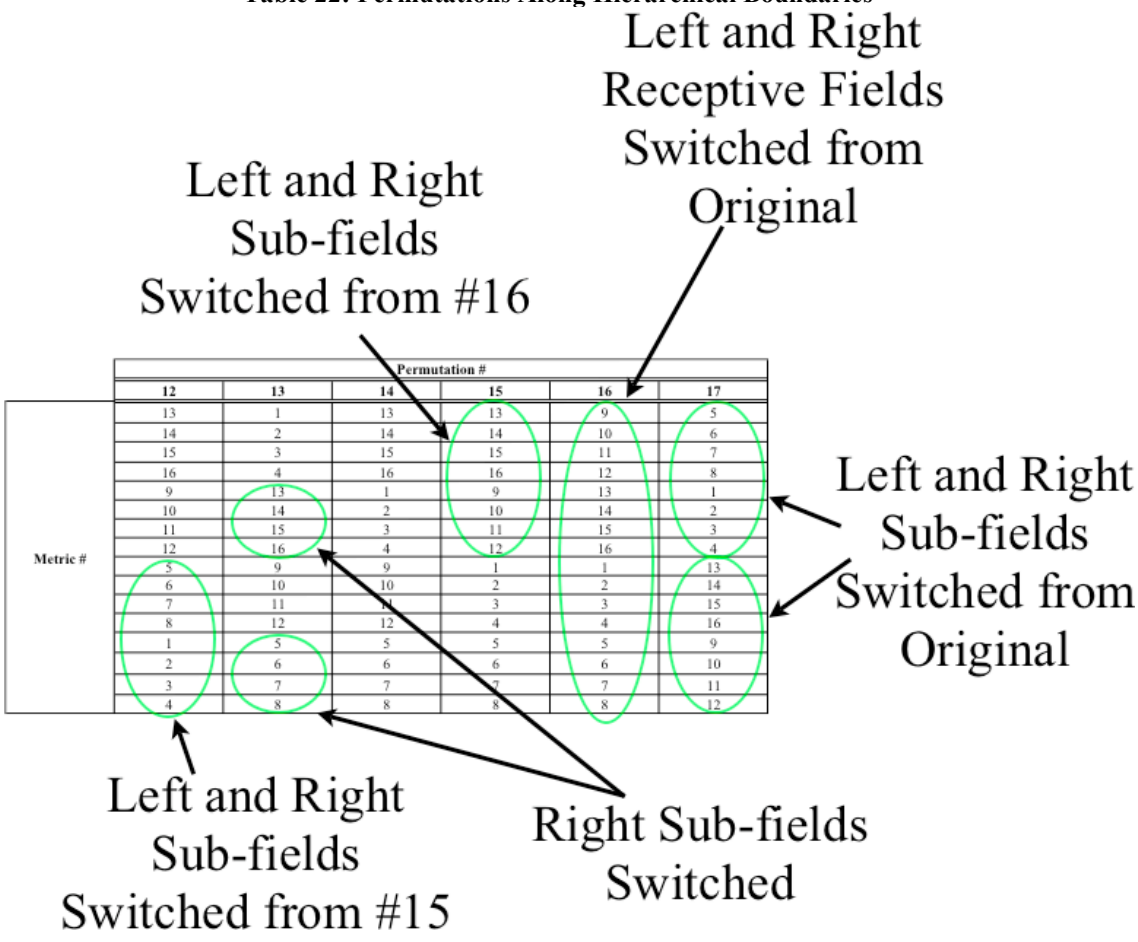
no surprise though because imagine the analogy with the Pictures Problem. For instance, if the vector components were randomly jumbled, then it would be quite difficult to learn objects from the images. Here though, the problem is in one-dimension and the order of the data is less constrained.

Second, we see that the original ordering of the metrics may have been well chosen. The original ordering of the data divided it among the sensors as follows: 2 security metrics and 2 surge-tracking metrics were fed to $N^{1,1}$, 4 surge-tracking metrics were fed to $N^{1,2}$, 4 surge-tracking metrics were fed to $N^{1,3}$, 4 politico-economic metrics were fed to $N^{1,4}$. The only conscious choice in this ordering had been to keep strongly related metrics within the same bottom-level receptive field. Consequently, it is possible that this lowered the amount of false positives learned by the network. Recall from earlier that the primary caveat to learning relationships from data is finding false positives [102]-[105]. But these permutations likely augment the chances of the network learning false positives from the data. Though further testing would be needed to verify this claim, such analysis is beyond our scope. Instead, we will end the discussion on random permutations here and turn to an exploration into how the data is hierarchically stored in the network.

Hierarchical Information Storage

The hierarchical storage of data in the network can be tested with a similar strategy, but now we permute the data along boundaries determined by the network topology. For instance, since the data has been fed into the sensors in groups of fours, we will maintain these groups. A set of permutation tests with this constraint in mind is shown below (Table 22).

Table 22: Permutations Along Hierarchical Boundaries

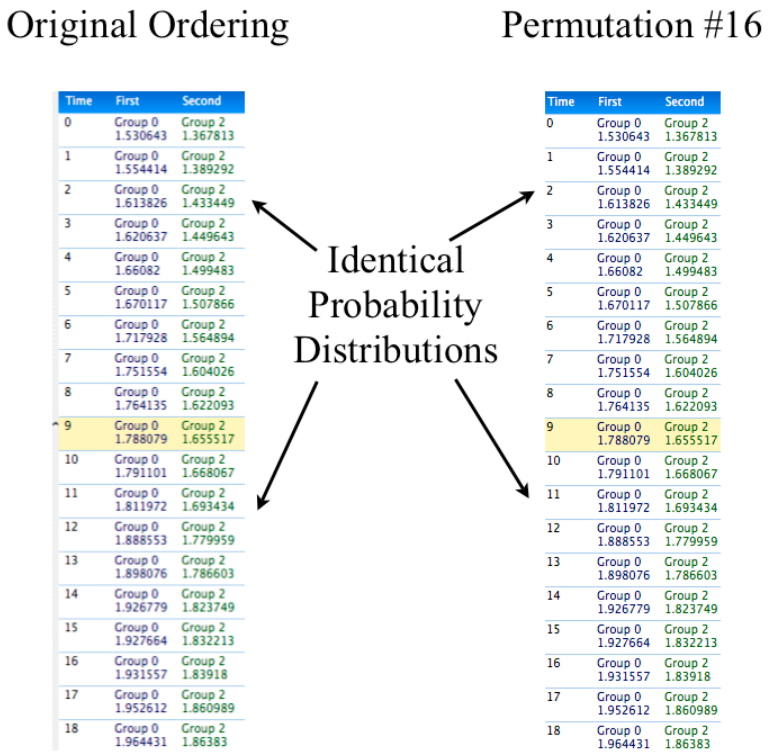


Metric #	12	13	14	15	16	17
	13	1	13	13	9	5
	14	2	14	14	10	6
	15	3	15	15	11	7
	16	4	16	16	12	8
	9	13	1	9	13	1
	10	14	2	10	14	2
	11	15	3	11	15	3
	12	16	4	12	16	4
	5	9	9	1	1	13
	6	10	10	2	2	14
	7	11	11	3	3	15
	8	12	12	4	4	16
	1	5	5	5	5	9
	2	6	6	6	6	10
	3	7	7	7	7	11
	4	8	8	8	8	12

We do not alter the network parameters from their original setup (C.9). Our over-arching goal with these permutations is to test the hierarchical storage of data in the nodes of the network. Do we have a schema related to politico-economic stability? Do we have a schema related to security metrics' indication of stability? Permutations of the data along some of these boundaries might give us some insight into these questions. For instance, permutation #16 switches the left and right receptive fields of the network. Specifically, metrics 1-8 are switched with 9-16. Also, permutation #17 switches within the left and right sub-fields. Specifically, subfields 1-4 are switched with 5-8 and 13-16 with 9-12. The other permutations can be seen in the table (Table 22). In this manner, we can study the hierarchical storage of the knowledge base concerning the Iraq context. So we can

potentially see if contributing aspects to stability are hierarchically combined to create an overall knowledge base on stability. The aim here is to see if the Iraq context network builds aspects to stability (e.g., politico-economic, security, etc.) the same that the Pictures context network builds edges and shapes into objects.

Let us examine training and inference on some of these permutations to find out. For instance, permutation #16 and the resulting network yield identical inference to what was seen from the original network. A comparison of an excerpt from the top-level node's inference history on real data for example can be seen below (Figure 64).



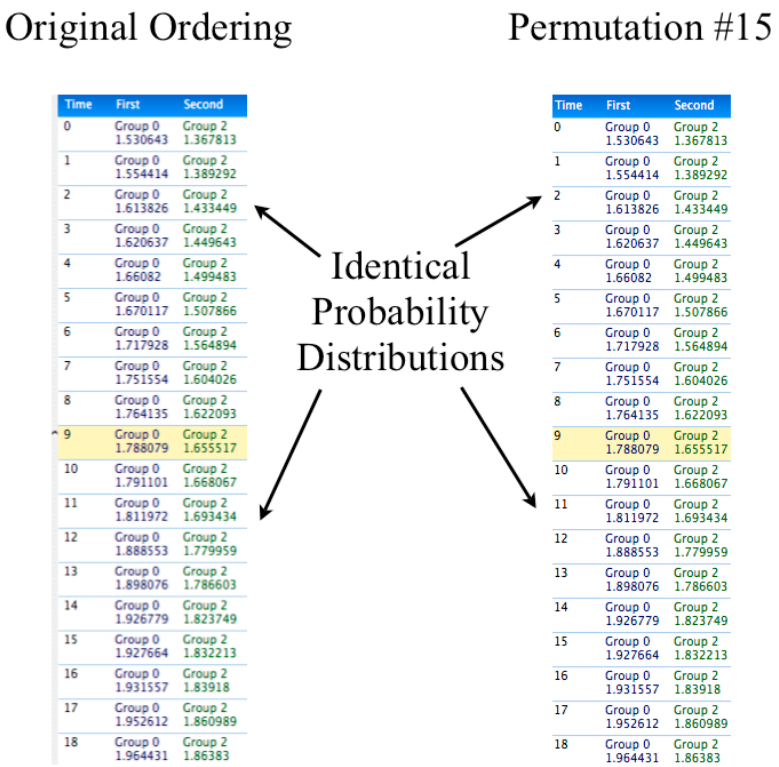
Time	First	Second
0	Group 0 1.530643	Group 2 1.367813
1	Group 0 1.554414	Group 2 1.389292
2	Group 0 1.613826	Group 2 1.433449
3	Group 0 1.620637	Group 2 1.449643
4	Group 0 1.66082	Group 2 1.499483
5	Group 0 1.670117	Group 2 1.507866
6	Group 0 1.717928	Group 2 1.564894
7	Group 0 1.751554	Group 2 1.604026
8	Group 0 1.764135	Group 2 1.622093
9	Group 0 1.788079	Group 2 1.655517
10	Group 0 1.791101	Group 2 1.668067
11	Group 0 1.811972	Group 2 1.693434
12	Group 0 1.888553	Group 2 1.779959
13	Group 0 1.898076	Group 2 1.786603
14	Group 0 1.926779	Group 2 1.823749
15	Group 0 1.927664	Group 2 1.832213
16	Group 0 1.931557	Group 2 1.83918
17	Group 0 1.952612	Group 2 1.860989
18	Group 0 1.964431	Group 2 1.86383

Time	First	Second
0	Group 0 1.530643	Group 2 1.367813
1	Group 0 1.554414	Group 2 1.389292
2	Group 0 1.613826	Group 2 1.433449
3	Group 0 1.620637	Group 2 1.449643
4	Group 0 1.66082	Group 2 1.499483
5	Group 0 1.670117	Group 2 1.507866
6	Group 0 1.717928	Group 2 1.564894
7	Group 0 1.751554	Group 2 1.604026
8	Group 0 1.764135	Group 2 1.622093
9	Group 0 1.788079	Group 2 1.655517
10	Group 0 1.791101	Group 2 1.668067
11	Group 0 1.811972	Group 2 1.693434
12	Group 0 1.888553	Group 2 1.779959
13	Group 0 1.898076	Group 2 1.786603
14	Group 0 1.926779	Group 2 1.823749
15	Group 0 1.927664	Group 2 1.832213
16	Group 0 1.931557	Group 2 1.83918
17	Group 0 1.952612	Group 2 1.860989
18	Group 0 1.964431	Group 2 1.86383

Figure 64: Comparison of Inference on Training Data – Second-Level Hierarchical Storage Proof

This trend continues for all values of t. This means that data is stored hierarchically through the second level of the network. If it were not then switching the left and right receptive fields would produce different top-level feed-forward inference. Another case is permutation #15. In this permutation, we start with permutation #16 and switch within

the left receptive field. So we switch the left and right sub-fields within the left receptive field. A comparison of an excerpt of the top-level node's inference history after training can be seen below (Figure 65).



Time	First	Second
0	Group 0 1.530643	Group 2 1.367813
1	Group 0 1.554414	Group 2 1.389292
2	Group 0 1.613826	Group 2 1.433449
3	Group 0 1.620637	Group 2 1.449643
4	Group 0 1.66082	Group 2 1.499483
5	Group 0 1.670117	Group 2 1.507866
6	Group 0 1.717928	Group 2 1.564894
7	Group 0 1.751554	Group 2 1.604026
8	Group 0 1.764135	Group 2 1.622093
9	Group 0 1.788079	Group 2 1.655517
10	Group 0 1.791101	Group 2 1.668067
11	Group 0 1.811972	Group 2 1.693434
12	Group 0 1.888553	Group 2 1.779959
13	Group 0 1.898076	Group 2 1.786603
14	Group 0 1.926779	Group 2 1.823749
15	Group 0 1.927664	Group 2 1.832213
16	Group 0 1.931557	Group 2 1.83918
17	Group 0 1.952612	Group 2 1.860989
18	Group 0 1.964431	Group 2 1.86383

Time	First	Second
0	Group 0 1.530643	Group 2 1.367813
1	Group 0 1.554414	Group 2 1.389292
2	Group 0 1.613826	Group 2 1.433449
3	Group 0 1.620637	Group 2 1.449643
4	Group 0 1.66082	Group 2 1.499483
5	Group 0 1.670117	Group 2 1.507866
6	Group 0 1.717928	Group 2 1.564894
7	Group 0 1.751554	Group 2 1.604026
8	Group 0 1.764135	Group 2 1.622093
9	Group 0 1.788079	Group 2 1.655517
10	Group 0 1.791101	Group 2 1.668067
11	Group 0 1.811972	Group 2 1.693434
12	Group 0 1.888553	Group 2 1.779959
13	Group 0 1.898076	Group 2 1.786603
14	Group 0 1.926779	Group 2 1.823749
15	Group 0 1.927664	Group 2 1.832213
16	Group 0 1.931557	Group 2 1.83918
17	Group 0 1.952612	Group 2 1.860989
18	Group 0 1.964431	Group 2 1.86383

Figure 65: Comparison of Inference on Training Data – Bottom-Level Hierarchical Storage Proof

As before, we see no change in the top-level inference output, indicating hierarchical storage of data within the $N^{1,1}$ and $N^{1,2}$ bottom-level nodes. This is a crucial result because it demonstrates hierarchical storage of politico-economic aspects to stability (for this network in $N^{1,1}$). It also does the same for four of the total ten metrics related to security. So we see here an analog to the Pictures Problem, in which pieces of the image were reconstructed into an overall object. Here, aspects of instability are reconstructed in an overall recognition of it. Permutation #12 starts from Permutation #15 and switches the sub-fields within the right receptive field. This also yields no changes in the top-level node's inference history on real or progressively stable/unstable data. Although the

categories are not as clear as they were for Permutation #15, this result also indicates hierarchical storage of certain aspects to stability. It is interesting to note that going through the same permutation tests with the network that trained on real data yields different results in the top-level node for Permutation #12. This is seen to be supporting evidence against that training method. But for the network here, trained on progressively stable/unstable data, there is no effect. Together, these permutations seem to be supporting evidence for the hierarchical condensation of data on the Iraq context with this network. So as long as the hierarchical boundaries are respected when switching the order of the metrics, there is no effect on the condensation of data.

There is a change in the way the data is learned and inferred once these hierarchical boundaries are violated. Consider Permutation #13 as an example of this. Here, within left and right receptive fields, the right sub-fields are switched. This yields the same number of coincidence patterns and Markov-chains in the top-level node. But the way the network recognizes bottom-up evidence differs during inference. For instance, the top-level inference on real data (C.17) is different from the original (C.11). Specifically, the progressively unstable state, *g1*, is most likely at $t = 11$ (as it was for the original in C.11). But no other time points are conclusively similar to what was seen in the original network. For instance, for $t \in \{[41,43], [47,49]\}$, *g1* is the second most likely Markov-chain, but that is seen nearly throughout the inference history. So we see here a permutation from the original that disrupts the hierarchical way by which data has been condensed. Specifically, two of the security metrics and two of the surge-tracking metrics have moved side-by-side with the politico-economic metrics. This creates different condensation of bottom-level output in the middle level. Consequently, the top-level output is slightly altered. Nevertheless, we still see some features of the original network's inference performance. So such permutations do not have a completely obliterating effect on the formed schema.

Summary of Tests

From reducing the number of metrics, changing how the data is fed in, and examining how the data is stored hierarchically, we can reach some conclusions about the effectiveness of the networks used here to enable a computational SA of the Iraq context. Some of these tests have revealed that significant progress has been made with this application, while others have shown that certain problems that still exist. For instance, reducing the number of metrics from $N = 16$ to $N = 8$ gives us similar results, implying that there is some structure to the data – real and training – that an HTM network is uncovering. Also, certain permutations of groups of metrics that preserved the hierarchy's boundaries showed the extent to which data on the Iraq context is stored hierarchically. But other permutations – for instance, the random ones – yielded different results. It had originally been presumed that these permutations would have no effect on learning/inference, but this hypothesis can be revised in light of the experimental evidence. Specifically, it seems that the HTM learns spatial profiles – visually seen as shapes in the Vitamin D Toolkit interface [129] – and then uses memory of these profiles for inference. So in effect, we do not really get away from the visual pattern recognition aspect to HTM. Rather, it seems that this is an integral part of the algorithms' operations. So any permutations of data that alter their spatial shapes necessarily create different SA of the context being analyzed.

Consequences of Experiment #2: Revisit System Dynamics Model in Level 1 SA

While the results about recognizing unstable states have been somewhat conclusive, there is still no way to recognize stable states because of how the progressively stable data was formed. Let us see if we can improve this by training the network with data that is more clearly asymptotically stable and unstable. In other words, let us revisit our system dynamics model (part of Level 1 SA) and remove the probabilistic logic gates that made some metrics oscillate in time. This time all metrics

will make a monotonic increase or decrease. So we will simplify the SD model dramatically now. Specifically, the factors affecting the Iraq context will linearly change the values of context metrics every time step. This is different from the probabilistic logic gates that had been used before. Now a five percent change in the value each time step is done, for both unstable and stable cases. Nationwide unemployment is slightly altered in the stable case so that the value does not drop too fast. This is implemented directly in the Excel spreadsheet without the macro. Everything else in the model (conservation laws, boundaries on certain metrics, etc.) is left unchanged. An example dataset can be found in an appendix (C.18). To differentiate this SD model from what we have already done, we will call these extreme-cases the 'monotonic extreme-cases' in order to differentiate them from the 'progressively extreme-cases'. With this new data, let us return to analysis of Level 2 SA to see the effects.

Level 2 SA: Training and Testing with Monotonic Extreme Cases

We proceed as we did before to train and test networks. Now though, we use data that is monotonically extreme and so we expect the network to learn to recognize clear progressions towards stability/instability. Let us take the same network used for the previous Level 2 SA (C.9). After training the network, we see that there are sixty-one coincidence patterns ($C^{3,1}$) and fifty-nine Markov-chains ($G^{3,1}$) in the top-level node. Compare this with the sixteen coincidence patterns and eight Markov-chains learned from the progressively stable/unstable data. There is not much difference. What does inference on the training data tell us though?

When we perform inference on training data, we now see a clear progression of Markov-chains as instability increases, but stability is still not clear. A complete history of the top-level node feed-forward inference can be found in an appendix (C.19). First, let us examine the progression towards instability. We see that each of the Markov-chains is distinct for each time step. The progression indicated by $\max\left[\lambda_t(g_r)\right]$ is *g0, g1, g2, ...,*

g58. Since we know the data is monotonic towards instability, we can reasonably claim that the Markov-chain labels are monotonic towards instability as well. For example, the bottom-up evidence when *g45* is most likely in the top level indicates a situation that is less stable than when *g5* is most likely.

When we look for instability gradations in the actual data, we see some interesting results (C.20). At *t = 11, 12*, the entire probability distribution over top-level Markov-chains shifts towards higher number Markov-chains. At *t = 11*, *g25, g24, g23* are in the top three (see Figure 66).

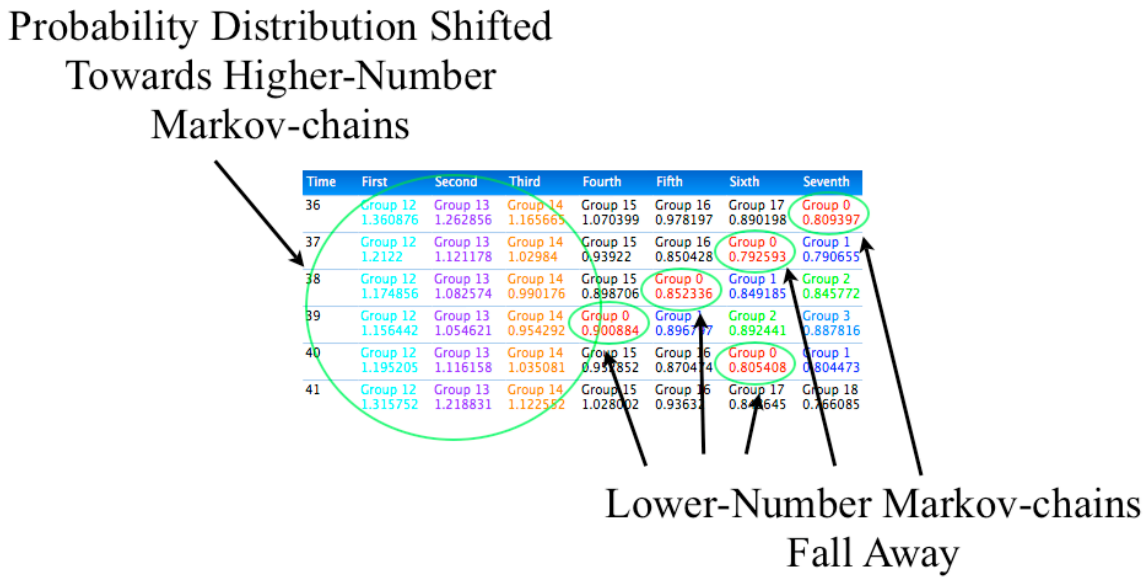
Probability Distribution Shifted Towards Higher-Number Markov-chains at t = 11, 12

Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh
11	Group 25 1.121081	Group 24 1.118762	Group 23 1.105853	Group 22 1.084392	Group 21 1.056441	Group 20 1.023943	Group 31 1.003414
12	Group 22 1.444566	Group 21 1.440598	Group 23 1.435091	Group 20 1.425162	Group 24 1.410539	Group 19 1.400369	Group 25 1.36988
13	Group 3 1.668496	Group 2 1.650038	Group 1 1.63119	Group 0 1.612177	Group 11 1.237494	Group 10 1.236002	Group 9 1.230558
14	Group 3 1.5965	Group 2 1.575433	Group 1 1.554296	Group 0 1.533289	Group 11 1.218317	Group 10 1.211122	Group 9 1.200339
15	Group 0 1.709698	Group 1 1.70499	Group 2 1.698328	Group 3 1.689481	Group 4 1.317045	Group 5 1.30309	Group 6 1.286214
16	Group 3 1.441241	Group 2 1.433847	Group 1 1.425541	Group 0 1.416527	Group 12 1.197943	Group 13 1.172897	Group 14 1.141887
17	Group 3 1.658548	Group 2 1.648333	Group 1 1.637012	Group 0 1.624839	Group 12 1.195943	Group 13 1.172231	Group 7 1.156083
18	Group 12 1.960125	Group 13 1.939044	Group 14 1.911954	Group 15 1.878536	Group 16 1.83861	Group 17 1.792183	Group 18 1.739495
19	Group 0 1.574832	Group 1 1.554279	Group 2 1.53177	Group 3 1.507229	Group 4 1.106257	Group 5 1.077488	Group 6 1.046577

Probability Distribution Shifted Towards Higher-Number Markov-chains at t = 18

Figure 66: Instability Recognition of Real Data

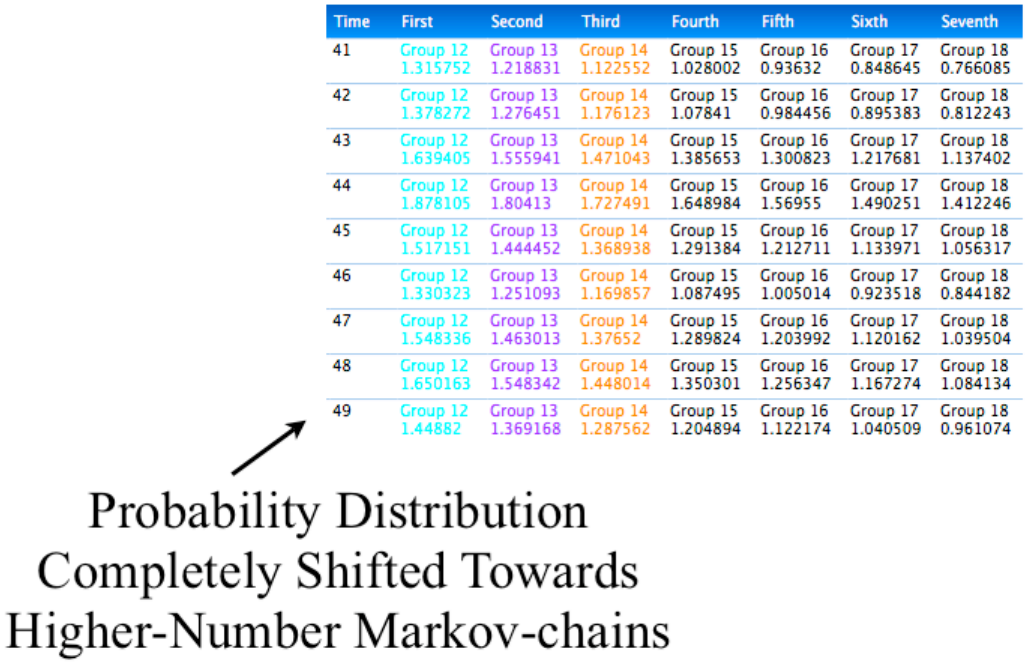
This is in line with what we have seen in previous networks at $t = 11$. Only now, we have a finer scale for grading instability. Rather than having one Markov-chain representing an "unstable" state, there is now a gradation between *g1* and *g58*. So the twenty-fifth Markov-chain being the most likely at this time point indicates the level of instability. At $t = 12$, *g22, g21, g23* are also in the top three. Now this is in line with a result also seen in a previous network's inference on training data (the network whose inference is shown in C.11), although it has not thus far been discussed. Why not? The reason for this is that the jump in the probability of the "unstable" state was not considered significant enough to warrant a discussion. But, in fact, the probability of *g1* – the tenth most likely Markov-chain – is one order of magnitude greater at $t = 12$ than it is for $t \in \{[6,10], [13,14]\}$ for that network (see Figure 79 for visual aid). Now though, with the monotonic-trained network, we see why. It is because we now have a better idea of how "unstable" this time point was: the twenty-second Markov-chain is most likely now, as opposed to the twenty-fifth at $t = 11$. In other words, the level of instability is not as great at $t = 12$ as it is at $t = 11$ because higher-number Markov-chains (e.g., *g25* compared to *g22*) indicate greater levels of instability. At $t = 18$, the probability distribution shifts as well, indicating *g12, g13, g14* in the top three. What is most interesting now is how the network recognizes the evidence for $t \in [41,49]$. Let us expand our purview to those time points leading up to and coming out of this time interval. If we consider the top seven Markov-chains of the top-level for $t \in [36,60]$ then we see something quite interesting. For $t \in [36,41]$, the distribution shifts increasingly towards *g12, g13, g14, g15, g16, g17, g18*. We can see this by the demotion of *g0* over these time steps too (Figure 67).



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh
36	Group 12 1.360876	Group 13 1.262856	Group 14 1.165665	Group 15 1.070399	Group 16 0.978197	Group 17 0.890198	Group 0 0.809397
37	Group 12 1.2122	Group 13 1.121178	Group 14 1.02984	Group 15 0.93922	Group 16 0.850428	Group 0 0.792593	Group 1 0.790655
38	Group 12 1.174856	Group 13 1.082574	Group 14 0.990176	Group 15 0.898706	Group 0 0.852336	Group 1 0.849185	Group 2 0.845772
39	Group 12 1.156442	Group 13 1.054621	Group 14 0.954292	Group 0 0.900884	Group 1 0.896777	Group 2 0.892441	Group 3 0.887816
40	Group 12 1.195205	Group 13 1.116158	Group 14 1.035081	Group 15 0.957852	Group 16 0.870474	Group 0 0.805408	Group 1 0.804473
41	Group 12 1.315752	Group 13 1.218831	Group 14 1.122582	Group 15 1.028002	Group 16 0.93631	Group 17 0.848645	Group 18 0.766085

Figure 67: Probability Distribution Shifts Towards Higher-Number Markov-chains

Then for $t \in [41,49]$, these seven Markov-chains are the most likely, given the bottom-up evidence (Figure 68).



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh
41	Group 12 1.315752	Group 13 1.218831	Group 14 1.122552	Group 15 1.028002	Group 16 0.93632	Group 17 0.848645	Group 18 0.766085
42	Group 12 1.378272	Group 13 1.276451	Group 14 1.176123	Group 15 1.07841	Group 16 0.984456	Group 17 0.895383	Group 18 0.812243
43	Group 12 1.639405	Group 13 1.555941	Group 14 1.471043	Group 15 1.385653	Group 16 1.300823	Group 17 1.217681	Group 18 1.137402
44	Group 12 1.878105	Group 13 1.80413	Group 14 1.727491	Group 15 1.648984	Group 16 1.56955	Group 17 1.490251	Group 18 1.412246
45	Group 12 1.517151	Group 13 1.444452	Group 14 1.368938	Group 15 1.291384	Group 16 1.212711	Group 17 1.133971	Group 18 1.056317
46	Group 12 1.330323	Group 13 1.251093	Group 14 1.169857	Group 15 1.087495	Group 16 1.005014	Group 17 0.923518	Group 18 0.844182
47	Group 12 1.548336	Group 13 1.463013	Group 14 1.37652	Group 15 1.289824	Group 16 1.203992	Group 17 1.120162	Group 18 1.039504
48	Group 12 1.650163	Group 13 1.548342	Group 14 1.448014	Group 15 1.350301	Group 16 1.256347	Group 17 1.167274	Group 18 1.084134
49	Group 12 1.44882	Group 13 1.369168	Group 14 1.287562	Group 15 1.204894	Group 16 1.122174	Group 17 1.040509	Group 18 0.961074

Figure 68: Complete Shift Towards Markov-chains Indicating Instability

As we know from previous discussions, there were peaks in violence and other attacks, sagging economic metrics, etc., that make it difficult to characterize the stability level during this time period. But here we see the probability distribution shift for the entire time period towards these mid-grade instable states. Then for $t \in [50,51]$, the probability distribution begins to shift back. Finally, for $t \in [52,60]$, *g0, g1, g2, g3* are the top four most likely Markov-chains (see Figure 69).

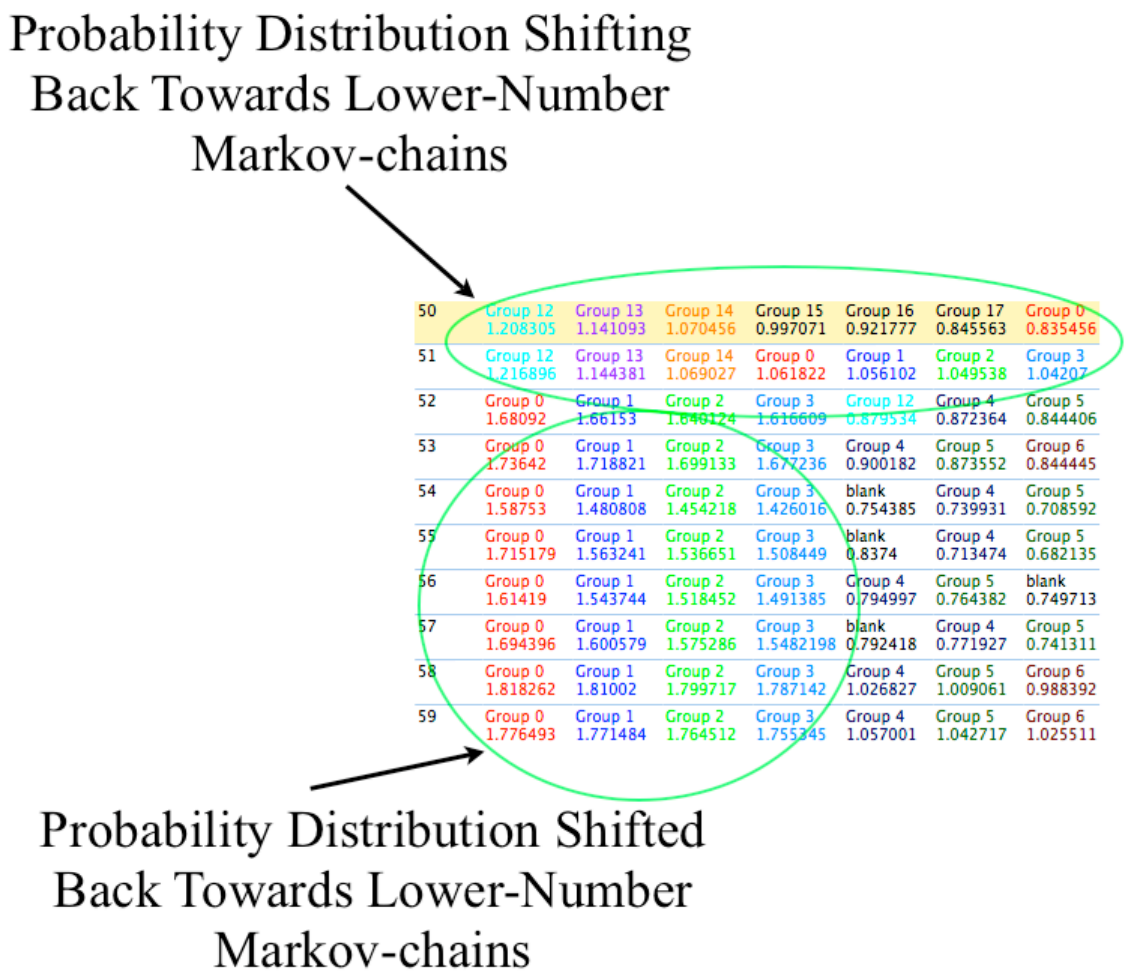


Figure 69: Complete Shift Back Towards Markov-chains Indicating Less Instability

Even though this does not indicate stability, it does indicate a dramatic drop in instability. So we see here that the monotonic training data has allowed us to more clearly analyze

when the evidence indicates trends towards instability. But what about stability recognition?

As mentioned earlier, stability recognition is less clear, even with the monotonic training data. Why is this so? If we consider the types of metrics used here then we notice that only four of them are proportional in value to stability. So, as a more stable situation is reached, the remaining twelve metrics drop close to zero. Consequently, the bottom-up evidence does not provide enough magnitude to propagate through the network. All the change comes from the four metrics proportional to stability. In the current permutation of the data, one of them is in the receptive field of $N^{1,3}$ and the other four are in the field of $N^{1,4}$. The entire left receptive field (covered by $N^{1,1}$ and $N^{1,1}$) therefore produces *blank* recognition. This is because there is simply not enough bottom-up evidence coming up through this side of the network. So as before, we are not able to determine gradations of stability, but now it is for another reason. Specifically, the utility function of these metrics is inversely proportional to stability. Consequently, as stability is reached, the magnitude of $\bar{e}_t$ goes to zero. It might be possible to alleviate this problem by transforming the data by an inverse, but this is not explored here. We have significant results already with instability recognition. Furthermore, these results imply stability recognition.

We note briefly here that to handle this problem a logarithm-transformation of the data was also attempted. Although this produced monotonic indication of Markov-chains during monotonically stable states, the inference performance on actual data was spotty. So in contrast to the shockwaves context, logarithm-transformation is not useful in the Iraq context for recognizing stability/instability patterns.

As before, we employ some additional tests to examine the results in more detail and we recall the techniques discussed earlier. So we examine: whether fewer metrics can be used to produce similar results; how results change if the data is fed in differently;

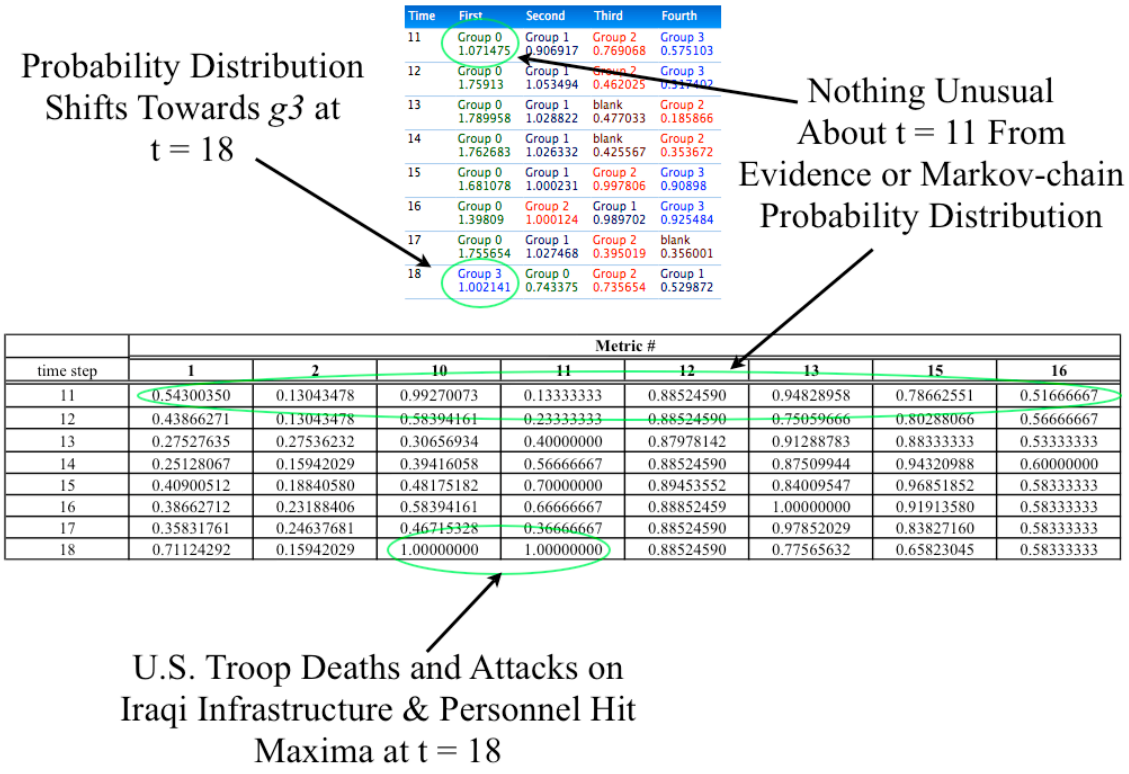
and, to what extent information is stored hierarchically. We will begin with an analysis of the effects from reducing the number of metrics.

Reduced Number of Metrics

As before, we reduce the number of metrics from $N = 16$ to $N = 8$. The same eight metrics are used here. The top-level node learns eight coincidence patterns ($C^{3,1}$) and four Markov-chains ($G^{3,1}$). When we look at inference on the training data, we see similar results to what we have seen earlier. The complete inference history is shown in an appendix (C.21). In particular, the monotonically stable states drive to zero and so produce *blank* Markov-chains that propagate up through the network. Also, the monotonic sequence of Markov-chains from $\max\left[\lambda_t(g_r)\right]$ reflects the monotonicity in the unstable training data. Specifically, this network recognizes the following sequence of Markov-chains via $\max\left[\lambda_t(g_r)\right]$: *g0, g1, g2, g3*.

When we perform inference on actual data, we see some interesting similarities to what was seen for the $N = 16$ case. For instance, at $t = 18$, *g3* is the most likely Markov-chain of the top-level node. This is what has happened in many other networks thus far. But it does not recognize anything in particular about $t = 11$. It seems that this occurs because the evidence was stronger at $t = 18$ (see Table 23).

Table 23: Effects of Reducing Number of Metrics



Time	First	Second	Third	Fourth
11	Group 0 1.071475	Group 1 0.906917	Group 2 0.769068	Group 3 0.575103
12	Group 0 1.75913	Group 1 1.053494	Group 2 0.462025	Group 3 0.347402
13	Group 0 1.789958	Group 1 1.028822	blank 0.477033	Group 2 0.185866
14	Group 0 1.762683	Group 1 1.026332	blank 0.425567	Group 2 0.353672
15	Group 0 1.681078	Group 1 1.000231	Group 2 0.997806	Group 3 0.90898
16	Group 0 1.39809	Group 2 1.000124	Group 1 0.989702	Group 3 0.925484
17	Group 0 1.755654	Group 1 1.027468	Group 2 0.395019	blank 0.356001
18	Group 3 1.002141	Group 0 0.743375	Group 2 0.735654	Group 1 0.529872

	Metric #							
time step	1	2	10	11	12	13	15	16
11	0.54300350	0.13043478	0.99270073	0.13333333	0.88524590	0.94828958	0.78662551	0.51666667
12	0.43866271	0.13043478	0.58394161	0.23333333	0.88524590	0.75059666	0.80288066	0.56666667
13	0.27527635	0.27536232	0.30656934	0.40000000	0.87978142	0.91288783	0.88333333	0.53333333
14	0.25128067	0.15942029	0.39416058	0.56666667	0.88524590	0.87509944	0.94320988	0.60000000
15	0.40900512	0.18840580	0.48175182	0.70000000	0.89453552	0.84009547	0.96851852	0.58333333
16	0.38662712	0.23188406	0.58394161	0.66666667	0.88852459	1.00000000	0.91913580	0.58333333
17	0.35831761	0.24637681	0.46715328	0.36666667	0.88524590	0.97852029	0.83827160	0.58333333
18	0.71124292	0.15942029	1.00000000	1.00000000	0.88524590	0.77565632	0.65823045	0.58333333

For instance, as the table shows, both U.S. troop deaths and attacks on Iraqi infrastructure/personnel hit a maximum at $t = 18$. Also, for $t \in [40,49]$, we see some interesting results that are similar to what has been seen before for $N = 16$. For $t \in [40,45]$, $\max\left[\lambda_t(g_r)\right]$ indicates $g1$. Then for $t \in [47,49]$, we see $\max\left[\lambda_t(g_r)\right]$ indicates $g2$ and $g1$. So this indicates, as before, that these time points have evidence indicating instability. This is similar to what was seen for the $N = 16$ network trained this way (see Figure 70).

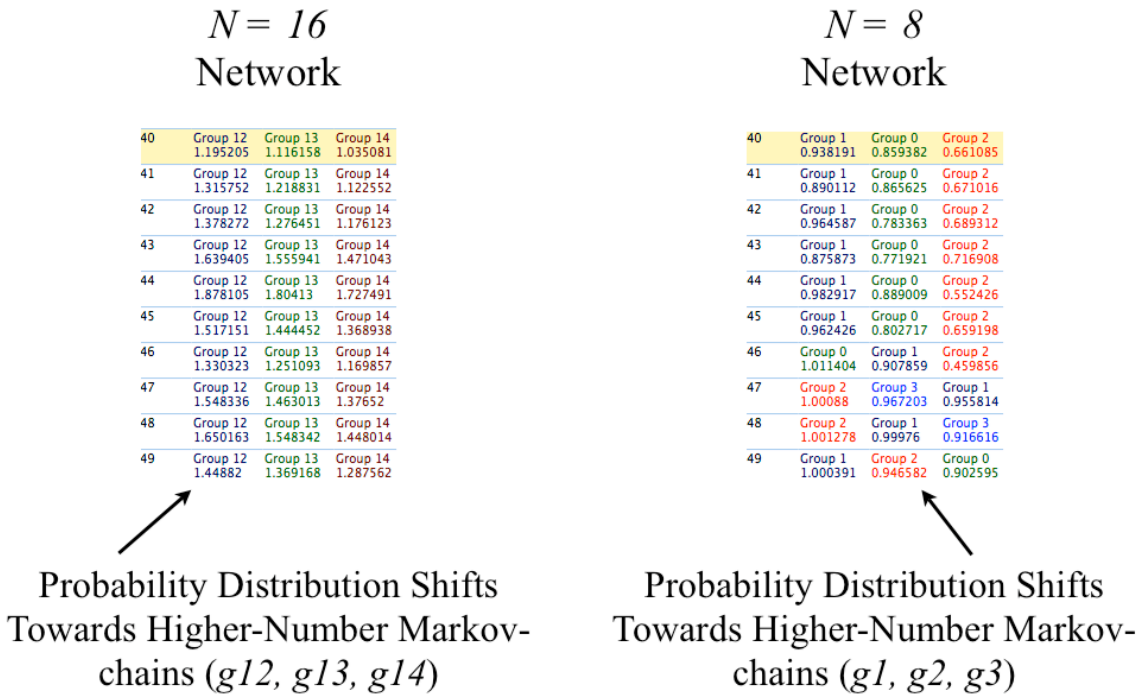


Figure 70: Comparison of *N = 16* and *N = 8* Inference - Both Recognize Heightened Instability

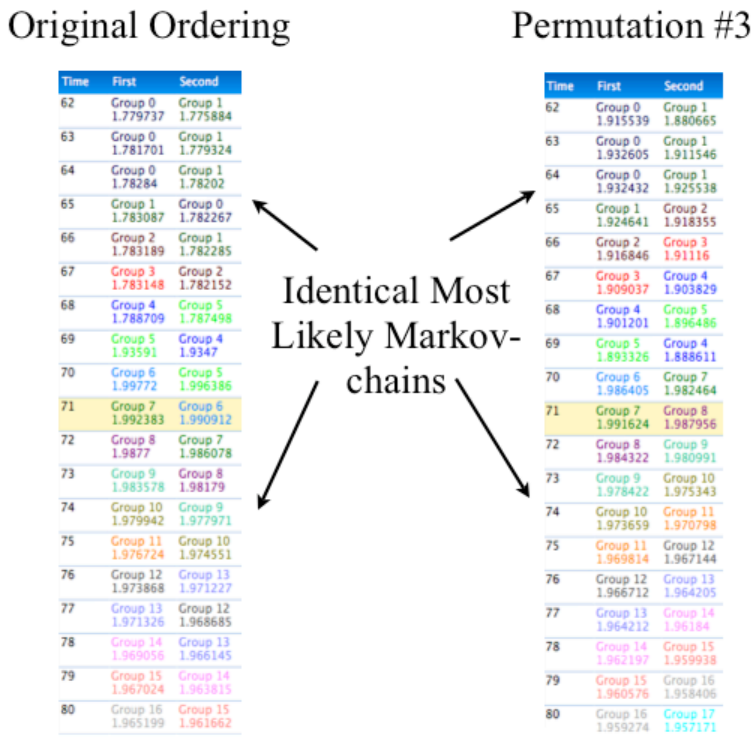
So we see that when we reduce the number of metrics, the recognition of "unstable" states is maintained when looking at the actual data. What about when we reduce the number of metrics further?

We reduce the number of metrics to $N = 4$ and assess training/inference results, having trained on the monotonic extreme-cases. The same four metrics that were used before are used here (Table 20). The top-level node learned four coincidence patterns ($C^{3,1}$) and three Markov-chains ($G^{3,1}$). When we look at the inference on the monotonic training data, we see encouraging results. The complete inference history can be found in an appendix (C.22). For instance, the monotonicity of groups indicated by $\max\left[\lambda_t(g_r)\right]$ is seen during stable states. We see that *g1* follows *g0* as the states become more stable. Also, the top-level node indicates a monotonic progression of Markov-chains for unstable states: *g2* follows *g0*. But when looking at inference on the real data, the results are inconclusive. This is a similar result to what was seen before when the number of metrics

was reduced below some threshold between $N = 4$ and $N = 8$ metrics. As it did before, this result demonstrates the need for a certain amount of metrics to be tracked when analyzing a complex system. Consequently, this is an instructive result for decision-makers operating within such contexts.

Alternative Ways of Feeding Data

As before, we test the effects of randomly permuting the data as it is fed into the network. So let us permute the metrics as we did before with Permutation #3 (Table 21). When we train the same network on the data, we get in the top-level node sixty coincidence patterns ($C^{3,1}$) and fifty-nine Markov-chains ($G^{3,1}$). This is only one fewer pattern than the original network's top-level node learned. When looking at inference on the training data though, we see an identical sequence of Markov-chains indicated by $\max\left[\lambda_t(g_r)\right]$ (see Figure 71).



Time	First	Second
62	Group 0 1.779737	Group 1 1.775884
63	Group 0 1.781701	Group 1 1.779324
64	Group 0 1.78284	Group 1 1.78202
65	Group 1 1.783087	Group 0 1.782267
66	Group 2 1.783189	Group 1 1.782285
67	Group 3 1.783148	Group 2 1.782152
68	Group 4 1.788709	Group 5 1.787498
69	Group 5 1.93591	Group 4 1.9347
70	Group 6 1.99772	Group 5 1.996386
71	Group 7 1.992383	Group 6 1.990912
72	Group 8 1.9877	Group 7 1.986078
73	Group 9 1.983578	Group 8 1.98179
74	Group 10 1.979942	Group 9 1.977971
75	Group 11 1.976724	Group 10 1.974551
76	Group 12 1.973868	Group 13 1.971227
77	Group 13 1.971326	Group 12 1.968685
78	Group 14 1.969056	Group 13 1.966145
79	Group 15 1.967024	Group 14 1.963815
80	Group 16 1.965199	Group 15 1.961662

Time	First	Second
62	Group 0 1.915539	Group 1 1.880665
63	Group 0 1.932605	Group 1 1.911546
64	Group 0 1.932432	Group 1 1.925538
65	Group 1 1.924641	Group 2 1.918355
66	Group 2 1.916846	Group 3 1.91116
67	Group 3 1.909037	Group 4 1.903829
68	Group 4 1.901201	Group 5 1.896486
69	Group 5 1.893326	Group 4 1.888611
70	Group 6 1.986405	Group 7 1.982464
71	Group 7 1.991624	Group 8 1.987956
72	Group 8 1.984322	Group 9 1.980991
73	Group 9 1.978422	Group 10 1.975343
74	Group 10 1.973659	Group 11 1.970798
75	Group 11 1.969814	Group 12 1.967144
76	Group 12 1.966712	Group 13 1.964205
77	Group 13 1.964212	Group 14 1.96184
78	Group 14 1.962197	Group 15 1.959938
79	Group 15 1.960576	Group 16 1.958406
80	Group 16 1.959274	Group 17 1.957171

Figure 71: Small Effects from Random Permutations of Metrics in Inference

Furthermore, the trend seen in the table continues for all values of $t \in [62, 122]$. This is an intriguing result because for the progressively stable/unstable permutations there were changes in this regard (e.g., compare this result to the comparable one between inference histories found in C.15 and C.10). Now though, there are none and we see that *g0* is recognized as most likely throughout the monotonically stable states. As before, during the monotonic unstable states, the progression of most likely Markov-chains is *g0, g1, ..., g58*. So the network recognizes a progression of increasingly unstable states. But as before, it cannot do so for the increasingly stable ones.

When we look at inference on real data, we see little change in the ability to recognize unstable states. A complete history is shown in an appendix for the top-level node (C.23). For instance, at $t \in \{11,18\}$, the probability distribution shifts towards higher Markov-chain numbers. This indicates increased instability at these time points, as it has for past observations. For $t \in \{41,43,47\}$, we see comparable shifts in the probability distribution that we have seen before. But, there are also some time points that we have identified in the past as "unstable" that are not recognized as such right now. For instance, $t = 49$ has been recognized as such by previous networks, although this network ranks *g24* to be only the seventh most likely Markov-chain here. So we see once again how the results can change slightly based on how the data is permuted when fed into the network. This is likely due to the different hierarchical condensation of data during learning. Overall though, the inference performance on real data bears strong resemblance to what has been seen earlier.

Other permutations of the data can be tested and similar results follow. The progression of recognized unstable states remains monotonic in the top-level node and similar time points are recognized to be closer to instability than others. In general, it seems that the monotonic training approach is less susceptible to random permutations of the data that violate the hierarchical boundaries imposed by the network. Let us continue

to investigate permutations of the data, though now we will permute the data along those boundaries.

Hierarchical Information Storage

As before, the hierarchical storage of data in the network can be tested with a similar permutation strategy. So we can use the permutations from earlier to investigate the extent to which the data is stored hierarchically in these networks. Let us use Permutation #16 first. Here we see identical top-level coincidence ($C^{3,1}$) and Markov-chain ($G^{3,1}$) learning from the original. Also, the inference performance on both data sets is identical at the top-level node. If we compare snapshots of inference on real data side by side then we can see an example of this (Figure 72) for $t \in [9,27]$.

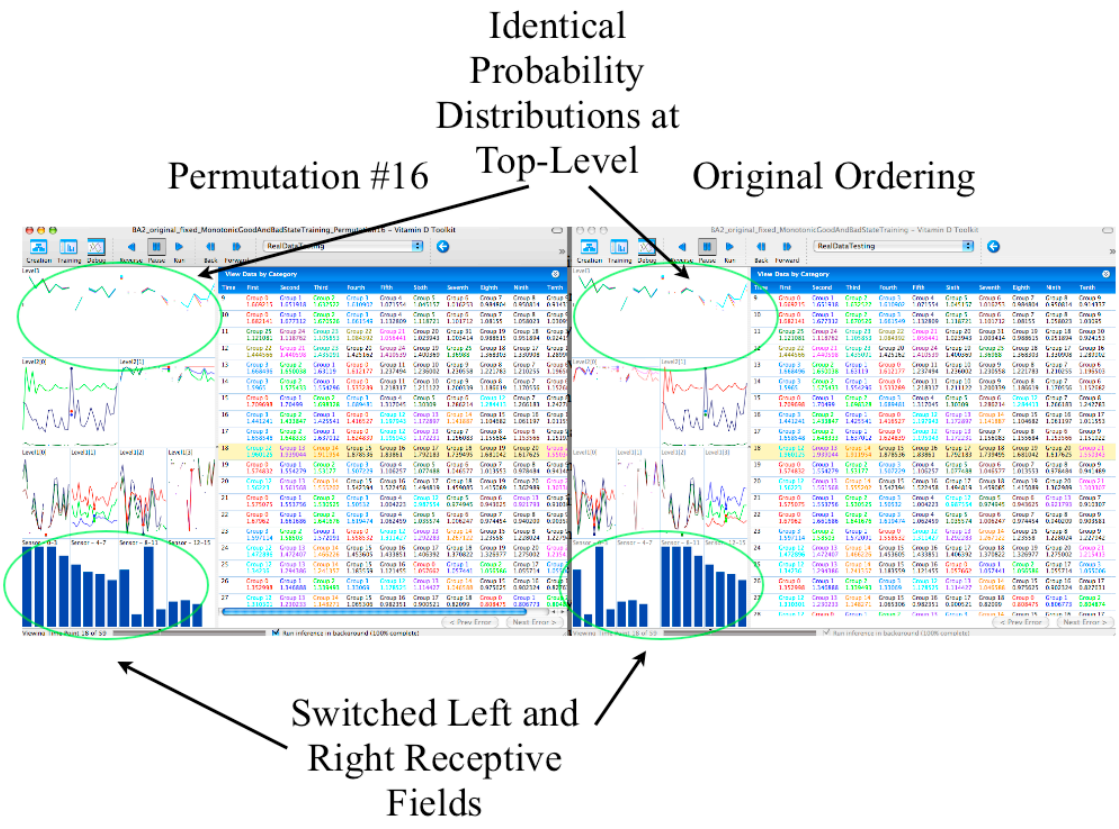


Figure 72: Comparison of Top-Level Node Probability Distributions - Identical for Hierarchical Permutations of Data

As before, this indicates that the data is stored hierarchically in the second level of the network. We find the same results when we do this test for Permutations #15, #12 and #17. This result for #15, #12 and #17 indicates hierarchical storage in the bottom-level nodes. Consequently, this means that aspects of instability are hierarchically stored and combined in the network, as it did earlier. Also as before, for Permutation #13, we see slight changes in the probabilities but the progression of most likely Markov-chains ($\max\left[\lambda_t\left(g_r\right)\right]$) is the same as with the original configuration of the data. Recall that Permutation #13 switches the right sub-fields within both left and right receptive fields. Consequently, this permutation violates the hierarchical boundaries between the metrics imposed by the network. Nevertheless, we see comparable inference performance from the top-level node on the actual data set. A comparison of inference excerpts can be seen below (Figure 73).

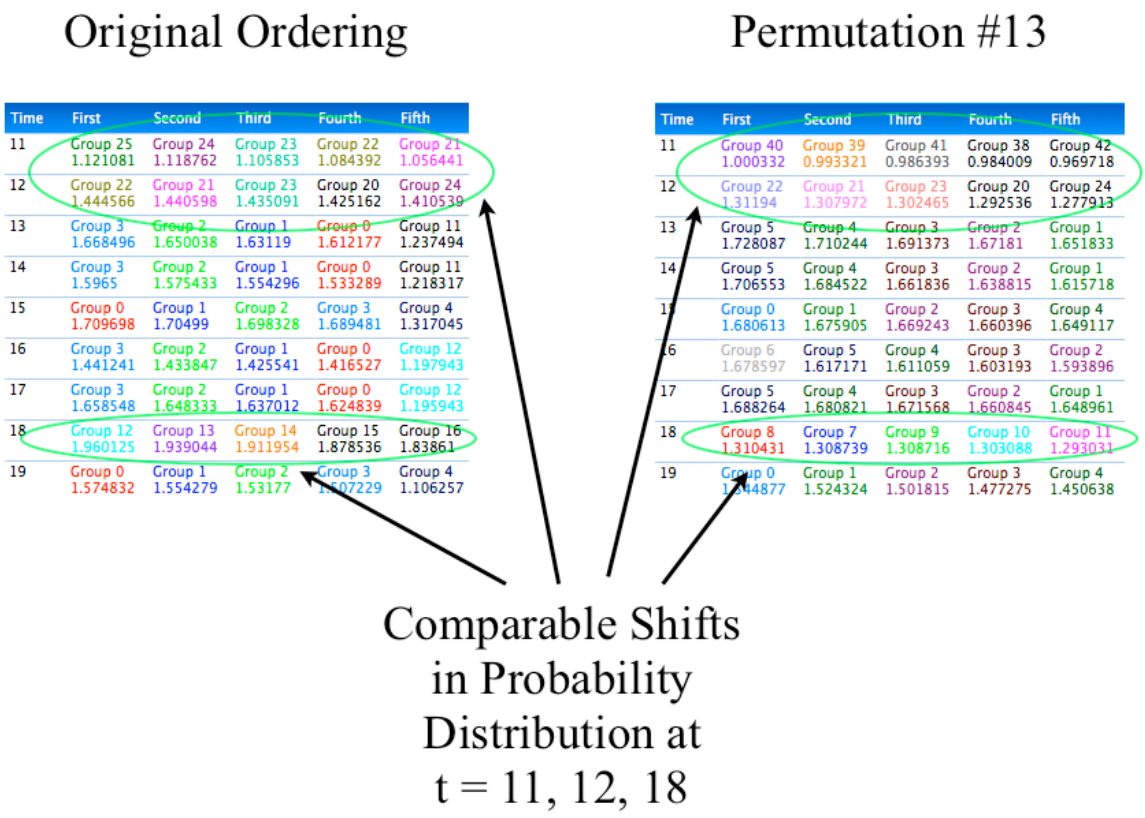


Figure 73: Comparable Probability Distribution Shifts Despite Permutation Violating Hierarchical Boundaries for Monotonically Trained Network

As the figure shows, at $t = 11, 12, 18$ there are shifts in the probability distribution towards higher number Markov-chains, indicating heightened instability. This is a familiar result from other networks, as the figure demonstrates. Furthermore, these comparisons continue for the rest of the inference history.

So we can conclude that, as before, there is little change in the network performance when the data is permuted in groups of four. The only slight change happens for Permutation #13, in which we switch sub-fields within left and right receptive fields. Since we have seen this effect twice, once for progressively trained and another time for monotonically trained networks, we can reasonably conclude that there is something about these metrics that makes this happen. Some possible reasons are that the permutation of metrics 5-8 with 13-16 isolates four of the politico-economic metrics within dissimilar other ones. For instance, crude oil production, crude oil export,

nationwide electricity and nationwide unemployment are now sandwiched between IED U.S. troop deaths and other hostile-fire U.S. troop deaths. Another possible – and related – reason is that three out of the four politico-economic metrics are proportional to stability, whereas none of the four metrics they replaced were. This means that during training on monotonic data, these subfields exhibit different end states. We saw before that these end states tend to zero. Consequently, this can also affect learning/inference.

Summary of Tests

These tests have revealed details about the training/testing approach done here. We have subjected the network trained on monotonic extreme-cases to these tests and have seen commendable performance. However, there are still many unknowns. Nevertheless, we see intriguing abilities for an HTM network to recognize instability via bottom-up evidence. We should recall that these abilities have been created via training the network on fictitious monotonic data. Then, when we analyze actual data, we are able to categorize the level of instability seen from the evidence. Quantitatively, this categorization is done with the Markov-chain probability distributions in the top-level.

Comparison to DoD Reports

Using pure data as bottom-up evidence, we have created Level 2 SA many times on the Iraq context. We have also seen some consistent results across a broad spectrum of tests and modifications. But how does this compare to what the DoD, whose Level 2 SA we are somewhat simulating, wrote about during this time? The reports on Iraq are quarterly and so they evolve over different intervals than the evidence [72]-[83][84]. Nevertheless, some comparison is possible to judge our networks' performance in light of these qualitative assessments. For instance, many of the networks examined here have identified fluctuating instability for $t \in \{41,43,47,49\}$. These time points correspond to

October 2006, December 2006, February 2007 and April 2007. Let us examine what the quarterly reports that cover these time periods describe about Iraq.

We begin with a comparison to the November 2006 report [77]. The executive summary of the report provides a good overview comparable to the one we have chosen by looking at $N = 16$ metrics to describe an entire complex system. Specifically, the summary focuses on security, political and economic aspects to the stability of Iraq. These are aspects we have considered as well. With regards to security, the DoD writes, "[Progress] is notable given the escalating violence in some of Iraq's more populous regions and the tragic loss of civilian life at the hands of terrorists and other extremists." If we look at $t = 41$, we see that some of this violence is reflected in the evidence (C.3). Consequently, previous networks indicated this time point as somewhat unstable. With regards to economic stability, the DoD writes, "[The] security situation, maintenance deficiencies, and management issues have adversely affected distribution and delivery of [essential services] ... Electrical distribution was affected by the same problems as the oil sector [.]" The data reflects these problems and many of the HTM networks pick up on this, for instance at $t = 41$. Of course, the problem is that the mapping between qualitative and quantitative description is never perfect. Nevertheless, these qualitative assessments do not contradict what we have seen thus far from quantitative analysis.

Comparison to the June 2007 report provides some interesting insight [79]. This report covers time points $t \in [45,48]$. Recall that $t \in \{47, 49\}$ have been generally identified by networks as somewhat unstable time points, so the overlap in time is not perfect but it will provide some comparison. The DoD writes as follows:

> The period covered in this report ... saw a greatly increased effort to secure turbulent areas ... some analysts see a growing fragmentation of Iraq ... Positive indicators include a decrease in civilian murders and sectarian violence ... while

negative indicators include the rise of high-profile attacks and expanded use of explosively formed projectiles.

So is this assessment conclusive? No, it is not at all. There are pieces of evidence indicating stability and other pieces indicating the opposite. So the reports do not contradict the HTM network findings, but they do not directly support them.

These two examples of comparative analysis demonstrate why we have shied away from it. This type of analysis is not as clear as testing the networks with extreme-case bounding. Although the reports do not contradict what has been seen in the networks' inference, they do not directly verify. Consequently, the use of extreme-case bounding seems to give a firmer indication about how well the Iraq context is understood with HTM.

Summary

In training networks on these two contexts, we have created a computational SA about them. For the shockwaves context, we saw that supervised networks provide the best SA when the goal was shock identification. Unsupervised networks were able to recognize distinct flow patterns and gave some indication – though not a unique one – about shocks. In the shockwaves context, four metrics described the whole context and so verification of our results was somewhat simple. We examined the results for when the number of metrics was dropped to $N = 2$ and saw some fall-off in unsupervised performance.

This test set the stage somewhat for the Iraq context. For this problem, the sixteen chosen metrics do not describe the whole context. After altering network parameters (discussed here only for the progressively extreme-case network), we used three techniques to test the SA formed about this context. These tests examined reduction in the number of metrics, altering the ordering of the data and examining the hierarchical

storage within the network. In general, the results were encouraging because different networks were able to recognize certain degrees of instability from the data. Also, although there is some comparative confirmation from DoD reports, the best guide has been the actual evidence and our training methods. Although this evidence does not describe the whole context, it covers many aspects of Iraq's stability that the DoD considers important. Consequently, there is noticeable overlap between the assessments given by the HTM networks and those of the DoD reports. But the analysis is more authoritative when done with extreme-case bounding, and so we lend more credence to that analytic approach for the Iraq context SA. **While the results are not perfect, we have definitely established and investigated a possible new way of analyzing complex system states, as exemplified by the Iraq context.**

CHAPTER 6

RESULTS

The previous chapter has showed how to form a degree of situational awareness (SA) about two contexts with HTM. The first one concerned the shockwaves context SA and it served as an extension from previous HTM work. The second context's SA was a primary goal from the literature review: stability assessment in Iraq. This context was meant to be an example of network-centric warfare and the challenge we have faced is how to form SA within this environment. Both of these implementations focused on Level 1 and Level 2 SA. So how would decision-making utilize this degree of SA in these contexts? Recall that the aim in using HTM for SA formation/maintenance has been to categorize situations to make decisions. Human factors researchers told us that categorization of a situation precedes decision-making and we have done that computationally in the previous chapter. Specifically, the top-level Markov-chains are the categories and the evidence-based probability distributions select certain ones over others. This is how we have computationally formed and tested SA. So let us now see how this SA can be utilized in light of operational goals. We follow the two contexts thus far used. These examples show that the analysis of a complex system is necessarily dependent on what information the observer/operator wishes to extract/utilize. In this chapter, we turn to step 6 of the research plan.

Using Shockwaves Context SA for Decision-Making

In the previous chapter, we found that certain HTM networks were able to recognize significant flow features. The significance of these features had been assumed because of their general applicability in aerospace engineering [111]-[118]. For instance,

sometimes it is important to recognize that a shock is about to occur because the properties will change discontinuously, possibly causing mechanical damage. Other times, it may be necessary to operate machinery within a range of temperature and pressure values. Whatever the situation, it is necessary to recognize the current condition of the flow and then react to it. Having the shockwaves context SA as an example, how would being able to recognize these flow features be of use?

Let us examine the case of the unsupervised log-transformed $N = 4$ network. This network recognized different flow patterns with commendable accuracy. So how can this knowledge be used for making decisions? Let us examine one of the perturbation cases. We can examine inference of the constant energy perturbation to the training data (see appendix B.18). Let us assume that a decision-maker operates something that depends on this context for which it is necessary to know when the flow pattern changes. But if he/she were using a Euclidean distance calculation, for example, to gauge pattern change, then his/her decisions would change every time step. We can see these changes by looking at a time excerpt of the log-transformed data (Table 24). But if he/she uses the output of a network (λ_t) trained on similar data, then he/she can adjust operation based on the Markov-chain that is most likely instead. For example, for $t \in [18,35]$, the flow properties change but the Markov-chain indicated by $\max\left[\lambda_t(g_r)\right]$ of the top-level node does not unless the flow does by a significant amount.

Table 24: Decision-Making Using Top-Level Feed-forward Inference vs. Euclidean Distance

Time	First	Second	Third	Euclidean distance between current and previous vector normalized units
18	Group 1 0.997451	Group 0 0.883761	Group 2 0.638287	0.039146176
19	Group 1 0.998841	Group 0 0.909097	Group 2 0.583054	0.043680614
20	Group 1 0.93039	Group 0 0.731216	Group 2 0.57796	0.165337905
21	Group 2 0.947616	Group 1 0.552662	Group 0 0.435291	0.456250327
22	Group 2 0.94554	Group 1 0.532581	Group 0 0.419933	0.015310705
23	Group 2 0.94842	Group 1 0.551718	Group 0 0.427662	0.026674044
24	Group 2 0.951587	Group 1 0.582561	Group 0 0.451709	0.021874386
25	Group 2 0.949017	Group 1 0.575088	Group 0 0.459772	0.035846058
26	Group 2 0.949378	Group 1 0.565233	Group 0 0.441414	0.025039183
27	Group 2 0.944872	Group 1 0.52755	Group 0 0.417274	0.034549647
28	Group 2 0.948351	Group 1 0.557813	Group 0 0.437676	0.026396993
29	Group 2 0.946007	Group 1 0.532229	Group 0 0.415748	0.018203352
30	Group 2 0.586433	Group 1 0.371437	Group 3 0.212452	0.410775515
31	Group 3 0.910645	Group 4 0.267907	Group 2 0.15229	0.831363649
32	Group 3 0.911121	Group 4 0.24322	Group 2 0.168034	0.03097663
33	Group 3 0.911143	Group 4 0.245723	Group 2 0.166684	0.004384709
34	Group 3 0.911979	Group 4 0.233015	Group 2 0.17905	0.016468678
35	Group 3 0.912595	Group 4 0.220927	Group 2 0.190907	0.015628862

Time

No Change in Flow Type

Oscillating Differences in Vectors

So the decision-maker can adjust operation according to the significant high-level features (in this case, *g1, g2* and *g3*) that are implicit in the data, rather than following insignificant noise. Similar rationale can be applied to other time intervals for which the probability of the r^{th} Markov-chain oscillates though the Markov-chain indicated by $\max\left[\lambda_t(g_r)\right]$ remains the same.

There is also some implicit knowledge about shocks within this network. Recall, this network experienced a drop in the value of the Markov-chain indicated by $\max\left[\lambda_t(g_r)\right]$ each time a shock appeared in the flow. Though supervision was required to instruct the network about the relationship between these events, a human user would see this pattern from the inference output. But we know from the way Level 1 SA was formed that each drop is physically significant. So if an operator monitors only the drop

in $\max\left[\lambda_t(g_r)\right]$ then he/she can also adjust decisions to account for the post-shock property changes. Of course, this assumes a response time that is on the order of shock transitions. But let us consider an example of a normal shock from Anderson ([114], p. 558) to see how much time this would approximately be. For $M_\infty = 12.28$, $T_\infty = 300K$ and $P_\infty = 1.8mmHG$, we can approximate the downstream flow speed and calculate a flow time between shock and equilibrium recovery to be about ~0.002 seconds. While this is a short amount of time for human reaction times, it is certainly within the realm of possibility for computerized response. So we see here a potential application for HTM-based flow pattern recognition in response to shocks. Of course, doing so would 'computerize' the decision-maker, since the required reaction time is so short. Similar applications could be done with the supervised HTM networks trained to recognize shocked flow. As the literature shows, there have been similar results elsewhere [127][128], but it is certainly a new method and perhaps one with a future.

Using Iraq Context SA for Decision-Making

In contrast to the shockwaves one, the Iraq context evolves on a longer time scale and is described with an incomplete set of metrics. How could the SA formed about this context be of use? Let us consider the networks trained with progressive and monotonic data. We saw general consistency in how these networks recognized varying degrees of instability. Of course, the recognition was not perfectly consistent, but the performance was commendable given the breadth of the problem. So how could a decision-maker use a network trained on such data to categorize the conditions in Iraq? Since categorization determines decisions, how would the decisions then be affected?

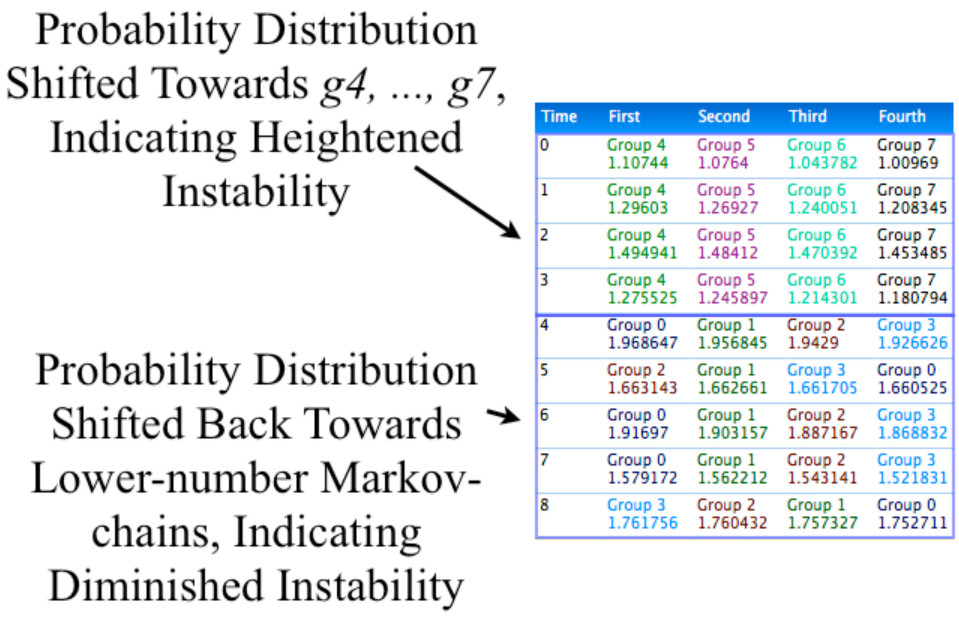
Decision-Making with a Monotonic Extreme-case Network

Let us consider the network trained on the $N = 16$ monotonic extreme-case data. Let us assume that we have a decision-maker who must react each month to the situation

in Iraq. The goal is to make Iraq more stable and so the decision-maker operates on the context to make this happen. For example, this is what the DoD as an organization has proclaimed it does [72]-[83]. We assume here that the decision-maker reacts somehow each month. But the exact operations of the decision-maker each month is not known. Then, an assessment with the SA formed about this context can then give the decision-maker feedback about the efficacy of the decision, whatever it might have been.

Since only data pertinent to this goal (i.e., stability) is considered in forming the network's knowledge base, the network provides a valuable tool in the control loop between the decision-maker and the context. For instance, consider the inference history of this network on the actual data (see appendix C.20). Once trained, this network has no temporal field of view beyond each month. So the network does not rely on the sequence of evidence to form an assessment. Rather, it propagates each month's evidence through the entire network and we can read out the λ_t of the top-level node for each month.

Let us simulate now how a decision-maker would utilize the monthly feed-forward inference. Consider λ_t for $t \in [0,8]$ (see Figure 74).



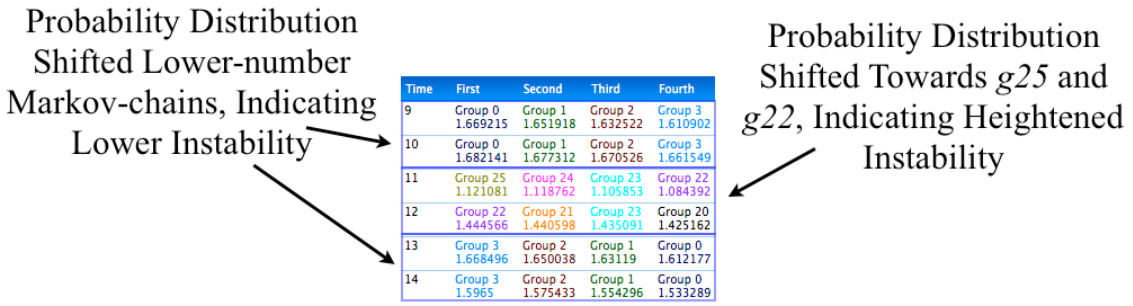
Time	First	Second	Third	Fourth
0	Group 4 1.10744	Group 5 1.0764	Group 6 1.043782	Group 7 1.00969
1	Group 4 1.29603	Group 5 1.26927	Group 6 1.240051	Group 7 1.208345
2	Group 4 1.494941	Group 5 1.48412	Group 6 1.470392	Group 7 1.453485
3	Group 4 1.275525	Group 5 1.245897	Group 6 1.214301	Group 7 1.180794
4	Group 0 1.968647	Group 1 1.956845	Group 2 1.9429	Group 3 1.926626
5	Group 2 1.663143	Group 1 1.662661	Group 3 1.661705	Group 0 1.660525
6	Group 0 1.91697	Group 1 1.903157	Group 2 1.887167	Group 3 1.868832
7	Group 0 1.579172	Group 1 1.562212	Group 2 1.543141	Group 3 1.521831
8	Group 3 1.761756	Group 2 1.760432	Group 1 1.757327	Group 0 1.752711

Figure 74: Using Top-Level Node's Feed-forward Inference for Decision-Making, *t ∈ [0,8]*

Here we see that *g4, g5, g6* and *g7* are the first through fourth most likely Markov-chains. If a decision-maker looks at this inference output for *t ∈ [0,3]*, then he/she has an indication that there are elements of the Iraq War that are somewhat unstable. This can then lead to decisions to try to augment stability. What these decisions would be is not the focus here, but as we know from human factors, the categorization of the situation is the first crucial step in making that decision. So instead we focus on the fact that the network allows the decision-maker to categorize the situation on a relative scale of instability. For *t ∈ [4,8]*, we see that the probability distribution shifts towards lower Markov-chains. Consequently, the decision-maker might take this information as an indication that current decisions are having the desired effect.

But decision-making, like schema formation, is a dynamic process. Specifically, decisions that work over a certain time period are not guaranteed to work over another. This is because the system evolves and so the actions required to drive it towards the goal

do too. Let us consider λ_t for $t \in [9,14]$ to see how a decision-loop would function in this regard (Figure 75).

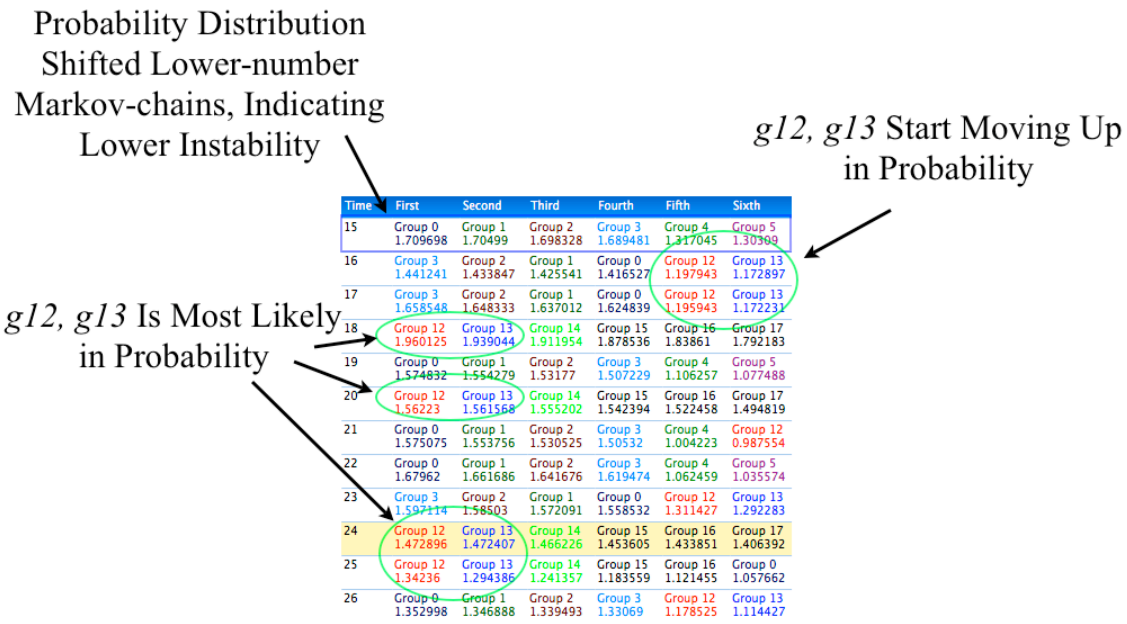


Time	First	Second	Third	Fourth
9	Group 0 1.669215	Group 1 1.651918	Group 2 1.632522	Group 3 1.610902
10	Group 0 1.682141	Group 1 1.677312	Group 2 1.670526	Group 3 1.661549
11	Group 25 1.121081	Group 24 1.118762	Group 23 1.105853	Group 22 1.084392
12	Group 22 1.444566	Group 21 1.440598	Group 23 1.435091	Group 20 1.425162
13	Group 3 1.668496	Group 2 1.650038	Group 1 1.63119	Group 0 1.612177
14	Group 3 1.5965	Group 2 1.575433	Group 1 1.554296	Group 0 1.533289

Figure 75: Using Top-Level Node's Feed-forward Inference for Decision-Making, *t ∈ [9,14]*

Here we see that the higher-number Markov-chains move up during $t \in [11,12]$. So our assumed decision-maker looks at λ_t for $t \in [9,10]$ and notices a probability distribution that is slanted towards less unstable states (e.g., *g0, g1, ...*). Consequently, operations continue as usual just after $t = 10$. But then the decision-maker looks at λ_t for $t = 11$ and notices a shift in the distribution towards higher-number Markov-chains. This shift indicates more instability than was seen for $t \in [9,10]$. Consequently, in pursuit of the goal to maintain stability, the decision-maker would then perform an operation (e.g., more troops) to drive λ_t towards that goal in the next time step. Let us assume that such action is taken, but then we get to $t = 12$ and the shift still exists. Then another operation can be attempted and λ_t re-evaluated. From the top-level node's inference of the Iraq context, we see that $t = 13$ began a shift away from instability. We do not claim here that we know *what* the exact operations are that are necessary to cause this shift. Rather, we claim here that it is possible to follow λ_t as a sort of universal metric to monitor stability in Iraq. By monitoring this universal metric, decisions can be made and their effects on the observed environment can be assessed.

We can continue in this fashion to simulate what a decision-maker would do with the feed-forward inference of an HTM. Let us consider $t \in [15,26]$ (see Figure 76).



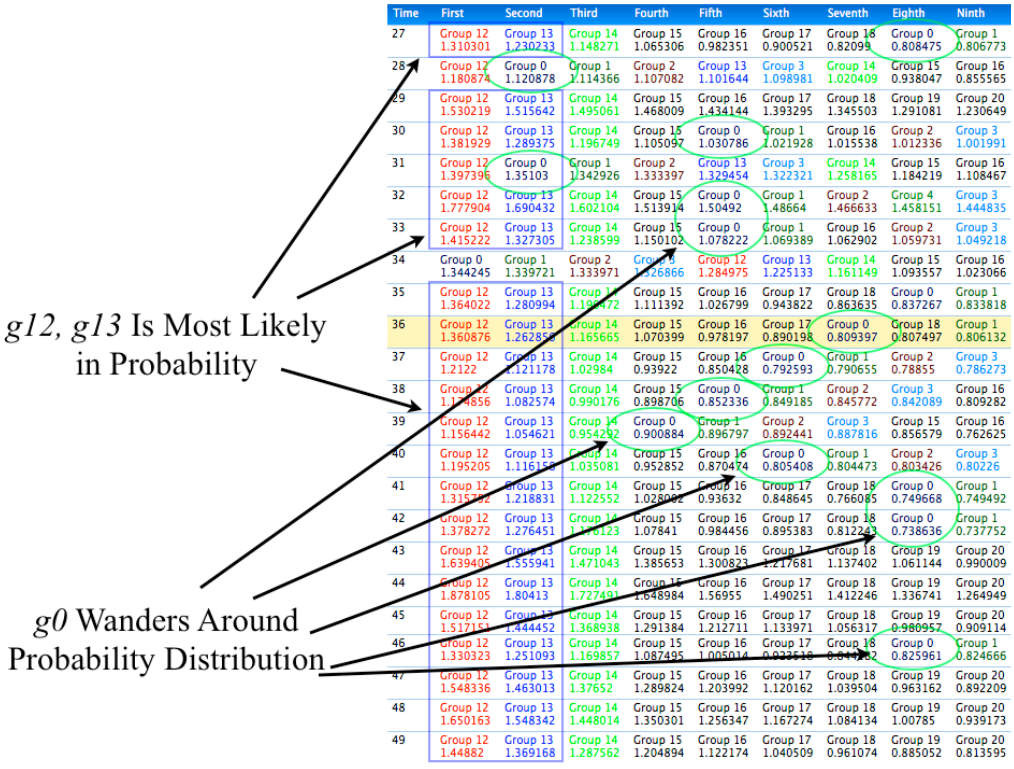
Time	First	Second	Third	Fourth	Fifth	Sixth
15	Group 0 1.709698	Group 1 1.70499	Group 2 1.698328	Group 3 1.689481	Group 4 1.317045	Group 5 1.30309
16	Group 3 1.441241	Group 2 1.433847	Group 1 1.425541	Group 0 1.416527	Group 12 1.197943	Group 13 1.172897
17	Group 3 1.658548	Group 2 1.648333	Group 1 1.637012	Group 0 1.624839	Group 12 1.195943	Group 13 1.172231
18	Group 12 1.960125	Group 13 1.939044	Group 14 1.911954	Group 15 1.878536	Group 16 1.83861	Group 17 1.792183
19	Group 0 1.574832	Group 1 1.554279	Group 2 1.53177	Group 3 1.507229	Group 4 1.106257	Group 5 1.077488
20	Group 12 1.56223	Group 13 1.561568	Group 14 1.555202	Group 15 1.542394	Group 16 1.522458	Group 17 1.494819
21	Group 0 1.575075	Group 1 1.553756	Group 2 1.530525	Group 3 1.50532	Group 4 1.004223	Group 12 0.987554
22	Group 0 1.67962	Group 1 1.661686	Group 2 1.641676	Group 3 1.619474	Group 4 1.062459	Group 5 1.035574
23	Group 3 1.597114	Group 2 1.58503	Group 1 1.572091	Group 0 1.558532	Group 12 1.311427	Group 13 1.292283
24	Group 12 1.472896	Group 13 1.472407	Group 14 1.466226	Group 15 1.453605	Group 16 1.433851	Group 17 1.406392
25	Group 12 1.34236	Group 13 1.294386	Group 14 1.241357	Group 15 1.183559	Group 16 1.121455	Group 0 1.057662
26	Group 0 1.352998	Group 1 1.346888	Group 2 1.339493	Group 3 1.33069	Group 12 1.178525	Group 13 1.114427

Figure 76: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [15,26]$

For $t = 15$, the instability level is relatively low. Specifically, there is a monotonic progression of Markov-chains in terms of likelihood at this month. So operations would continue as usual. But then for $t \in [16,17]$, something starts to change. Specifically, the probabilities of *g12* and *g13* move up to the top seven. Furthermore, $\max\left[\lambda_t(g_r)\right]$ indicates *g4* on this interval. So the decision-maker could take this information as an indication that there are elements to instability emerging from the evidence. First of all, *g4* is the most likely Markov-chain, so instability has risen, based on our monotonically trained scale. Second of all, the rise in *g12* and *g13* indicates an increased probability that the situation is more reminiscent of greater degrees of instability. This is not a prediction mechanism *per se*, but a decision-maker could use the lower probabilities of the distribution as an indication about the likelihood of those Markov-chains (i.e., instability levels) arising in the future. Let us assume that operations proceeded in such a way that we arrive at $t = 18$. Here there is a shift in the probability distribution towards *g12, g13*, etc. So it is possible that the upward shift of these Markov-chains for $t \in [16,17]$ was

somewhat indicative of what was about to happen. Of course, there had been no guarantee – only an increased likelihood. Perhaps operations change in such a way that at $t = 19$ the probability distribution shifts back towards lower-number Markov-chains. We do not know for sure, but we can get the feeling from this simulation how a decision-maker could use the information output by this network to guide decisions on the Iraq War.

As we go to later time periods, we see a persistent monthly recognition of high instability states. For instance, for $t \in [27,49]$ (Figure 77) the oscillation in the top end of the probability distribution subsides.



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
27	Group 12 1.310301	Group 13 1.230233	Group 14 1.148271	Group 15 1.065306	Group 16 0.982351	Group 17 0.900521	Group 18 0.82099	Group 0 0.808475	Group 1 0.806773
28	Group 12 1.180874	Group 0 1.120878	Group 1 1.114366	Group 2 1.107082	Group 13 1.101644	Group 3 1.098981	Group 14 1.020409	Group 15 0.938047	Group 16 0.855565
29	Group 12 1.530219	Group 13 1.515642	Group 14 1.495061	Group 15 1.468009	Group 16 1.434144	Group 17 1.393295	Group 18 1.345503	Group 19 1.291081	Group 20 1.230649
30	Group 12 1.381929	Group 13 1.289375	Group 14 1.196749	Group 15 1.105097	Group 0 1.030786	Group 1 1.021928	Group 16 1.015538	Group 2 1.012336	Group 3 1.001991
31	Group 12 1.397396	Group 0 1.35103	Group 1 1.342926	Group 2 1.333397	Group 13 1.329454	Group 3 1.322321	Group 14 1.258165	Group 15 1.184219	Group 16 1.108467
32	Group 12 1.777904	Group 13 1.690432	Group 14 1.602104	Group 15 1.513914	Group 0 1.50492	Group 1 1.48664	Group 2 1.466633	Group 4 1.458151	Group 3 1.444835
33	Group 12 1.415222	Group 13 1.327305	Group 14 1.238599	Group 15 1.150102	Group 0 1.078222	Group 1 1.069389	Group 16 1.062902	Group 2 1.059731	Group 3 1.049218
34	Group 0 1.344245	Group 1 1.339721	Group 2 1.333971	Group 3 1.326866	Group 12 1.284975	Group 13 1.225133	Group 14 1.161149	Group 15 1.093557	Group 16 1.023066
35	Group 12 1.364022	Group 13 1.280994	Group 14 1.196872	Group 15 1.111392	Group 16 1.026799	Group 17 0.943822	Group 18 0.863635	Group 0 0.837267	Group 1 0.833818
36	Group 12 1.360876	Group 13 1.262858	Group 14 1.165665	Group 15 1.070399	Group 16 0.978197	Group 17 0.890198	Group 0 0.809397	Group 18 0.807497	Group 1 0.806132
37	Group 12 1.2122	Group 13 1.121178	Group 14 1.02984	Group 15 0.93922	Group 16 0.850428	Group 0 0.792593	Group 1 0.790655	Group 2 0.78855	Group 3 0.786273
38	Group 12 1.174856	Group 13 1.082574	Group 14 0.990176	Group 15 0.898706	Group 0 0.852336	Group 1 0.849185	Group 2 0.845772	Group 3 0.842089	Group 16 0.809282
39	Group 12 1.156442	Group 13 1.054621	Group 14 0.954292	Group 0 0.900884	Group 1 0.896797	Group 2 0.892441	Group 3 0.887816	Group 15 0.856579	Group 16 0.762625
40	Group 12 1.195205	Group 13 1.116158	Group 14 1.035081	Group 15 0.952852	Group 16 0.870474	Group 0 0.805408	Group 1 0.804473	Group 2 0.803426	Group 3 0.80226
41	Group 12 1.315782	Group 13 1.218831	Group 14 1.122552	Group 15 1.028082	Group 16 0.93632	Group 17 0.848645	Group 18 0.766085	Group 0 0.749668	Group 1 0.749492
42	Group 12 1.378272	Group 13 1.276451	Group 14 1.176123	Group 15 1.07841	Group 16 0.984456	Group 17 0.895383	Group 18 0.812243	Group 0 0.738636	Group 1 0.737752
43	Group 12 1.639405	Group 13 1.555941	Group 14 1.471043	Group 15 1.385653	Group 16 1.300823	Group 17 1.217681	Group 18 1.137402	Group 19 1.061144	Group 20 0.990009
44	Group 12 1.878105	Group 13 1.80413	Group 14 1.727491	Group 15 1.648984	Group 16 1.56955	Group 17 1.490251	Group 18 1.412246	Group 19 1.336741	Group 20 1.264949
45	Group 12 1.517151	Group 13 1.444452	Group 14 1.368938	Group 15 1.291384	Group 16 1.212711	Group 17 1.133971	Group 18 1.056317	Group 19 0.980957	Group 20 0.909114
46	Group 12 1.330323	Group 13 1.251093	Group 14 1.169857	Group 15 1.087495	Group 16 1.005014	Group 17 0.923518	Group 18 0.844282	Group 0 0.825961	Group 1 0.824666
47	Group 12 1.548336	Group 13 1.463013	Group 14 1.37652	Group 15 1.289824	Group 16 1.203992	Group 17 1.120162	Group 18 1.039504	Group 19 0.963162	Group 20 0.892209
48	Group 12 1.650163	Group 13 1.548342	Group 14 1.448014	Group 15 1.350301	Group 16 1.256347	Group 17 1.167274	Group 18 1.084134	Group 19 1.00785	Group 20 0.939173
49	Group 12 1.44882	Group 13 1.369168	Group 14 1.287562	Group 15 1.204894	Group 16 1.122174	Group 17 1.040509	Group 18 0.961074	Group 19 0.885052	Group 20 0.813595

Figure 77: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [27,49]$

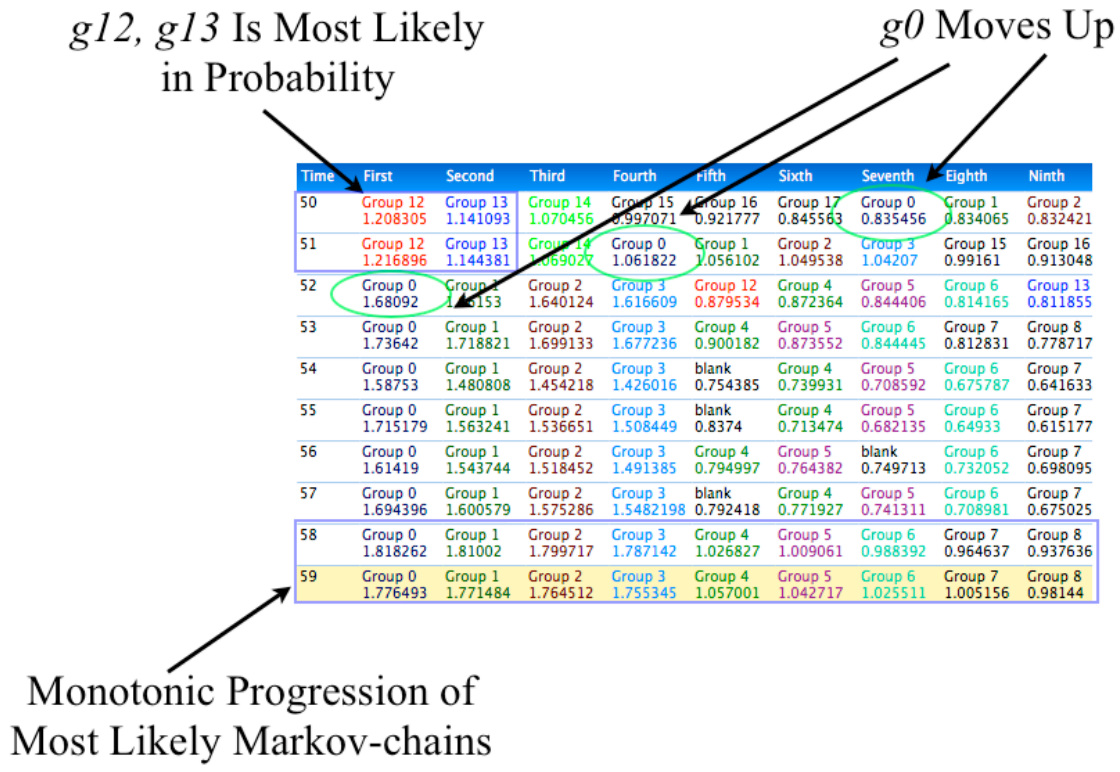
Each month (except for $t = 34$), the decision-maker would see that $\max\left[\lambda_t(g_r)\right]$ indicates *g12*. For $t \in [27,49]$, *g0* seems to wander around the probability distribution. Sometimes it is fifth, third or fourth most likely. So it is clear that the decisions that would bring Iraq

closer to stability are not happening at each of these time points. This seems to compare well with what the DoD reports say about part of this time period [74]. For instance, one report lays out the following:

> During this reporting period, the President of the United States, acting upon the recommendations of military commanders, authorized an adjustment to the U.S. force posture in Iraq, decreasing the number of combat brigades in Iraq from 17 to 15, a reduction of about 7,000 troops. This decision was based on several indicators of progress but primarily the growing capability of Iraqi Security Forces.

As before, there is not perfect overlap in terms of the time interval of these reports (quarterly) and of our data (monthly). Nevertheless, this assessment applies to $t \in [29,33]$ and so it is relevant to discuss it here. We see in this statement a point-blank acknowledgement that the decision to reduce U.S. forces in Iraq was based only on a handful of "indicators of progress." Primarily though it was based on the growing capability of the ISF. In other words, the decision had been based more on the potential – or capability – for progress, rather than the actuality of it. As before, the DoD reports do not conflict with our network's assessment of the stability of Iraq during this time period. Nor do the reports confirm the assessment. But we see here a cause for the floundering stability witnessed from the network's evidence-based inference during this and other time periods. The network only reacts to evidence, not capabilities, and so there is no discord between the two assessments.

It is not until March 2007 that we start to see the effects of change in the decisions regarding Iraq's stability. Consider $t \in [50,59]$ and note how the decision-maker would proceed through each month (Figure 78).



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
50	Group 12 1.208305	Group 13 1.141093	Group 14 1.070456	Group 15 0.997071	Group 16 0.921777	Group 17 0.845563	Group 0 0.835456	Group 1 0.834065	Group 2 0.832421
51	Group 12 1.216896	Group 13 1.144381	Group 14 1.069027	Group 0 1.061822	Group 1 1.056102	Group 2 1.049538	Group 3 1.04207	Group 15 0.99161	Group 16 0.913048
52	Group 0 1.68092	Group 1 1.65153	Group 2 1.640124	Group 3 1.616609	Group 12 0.879534	Group 4 0.872364	Group 5 0.844406	Group 6 0.814165	Group 13 0.811855
53	Group 0 1.73642	Group 1 1.718821	Group 2 1.699133	Group 3 1.677236	Group 4 0.900182	Group 5 0.873552	Group 6 0.844445	Group 7 0.812831	Group 8 0.778717
54	Group 0 1.58753	Group 1 1.480808	Group 2 1.454218	Group 3 1.426016	blank 0.754385	Group 4 0.739931	Group 5 0.708592	Group 6 0.675787	Group 7 0.641633
55	Group 0 1.715179	Group 1 1.563241	Group 2 1.536651	Group 3 1.508449	blank 0.8374	Group 4 0.713474	Group 5 0.682135	Group 6 0.64933	Group 7 0.615177
56	Group 0 1.61419	Group 1 1.543744	Group 2 1.518452	Group 3 1.491385	Group 4 0.794997	Group 5 0.764382	blank 0.749713	Group 6 0.732052	Group 7 0.698095
57	Group 0 1.694396	Group 1 1.600579	Group 2 1.575286	Group 3 1.5482198	blank 0.792418	Group 4 0.771927	Group 5 0.741311	Group 6 0.708981	Group 7 0.675025
58	Group 0 1.818262	Group 1 1.81002	Group 2 1.799717	Group 3 1.787142	Group 4 1.026827	Group 5 1.009061	Group 6 0.988392	Group 7 0.964637	Group 8 0.937636
59	Group 0 1.776493	Group 1 1.771484	Group 2 1.764512	Group 3 1.755345	Group 4 1.057001	Group 5 1.042717	Group 6 1.025511	Group 7 1.005156	Group 8 0.98144

Figure 78: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [50,59]$

At $t = 50$, the decision-maker notes that his decisions create an increase in the probability of *g0*. So he/she continues operations this way. Then at $t = 51$, *g0* has jumped again, in the process demoting other gradations of instability. Then at $t = 52$, the entire probability distribution has shifted, with the still persistent presence of *g12* as the fifth most likely Markov-chain. But the decision-maker sees the shift in probability distribution and continues operations. For each month of $t \in [53,59]$, the decision-maker notes his/her actions in light of the probability distribution that remains slanted towards lower-number Markov-chains. Consequently, he/she can see the desirable nature of his/her actions in light of established goals.

This is somewhat in line with what actually happened when President Bush implemented *The New Way Forward* in Iraq. This new approach to stabilizing Iraq was announced in January 2007 ($t = 44$) and was phased in through May 2007 ($t = 48$). As

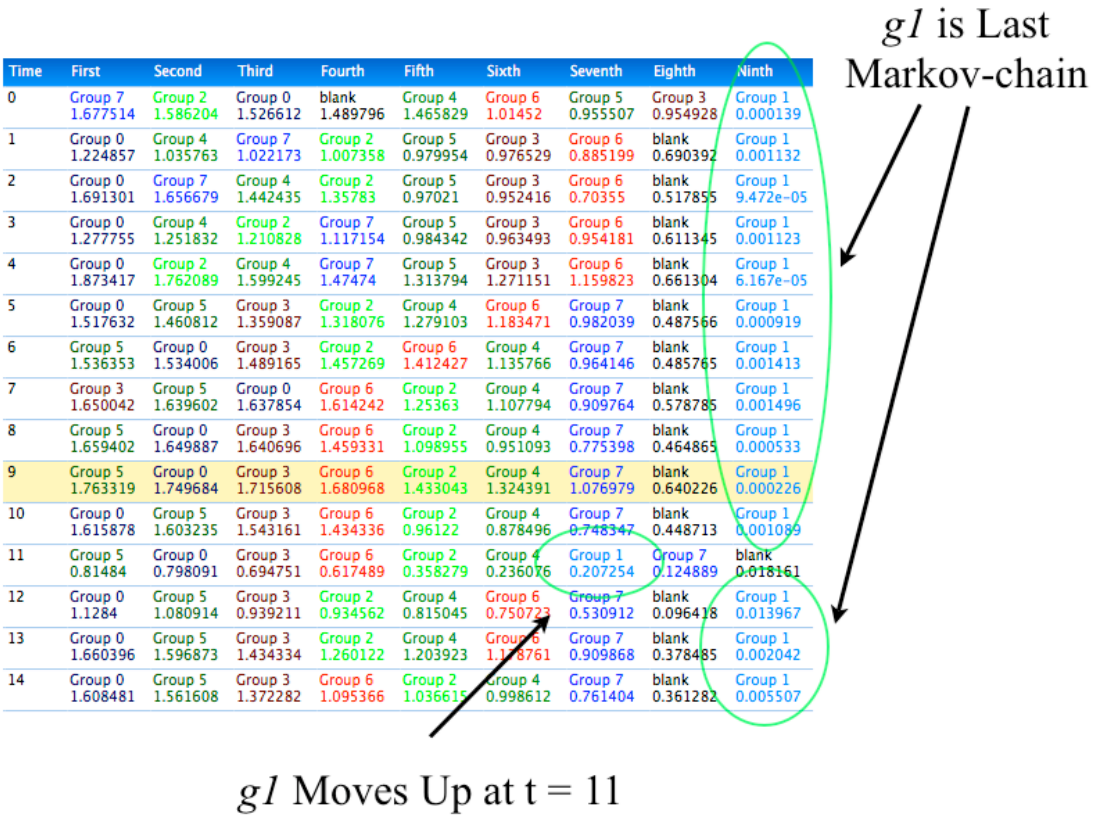
our bottom-up evidence shows (see appendix C.3), troop levels remained high through $t = 59$. But this one metric out of the $N = 16$ is not what is steering the top-level node's inference. Rather, other metrics are contributing to the λ_t output by the network. This is guaranteed because of how we have normalized all metrics according to the same rule. So no one metric is more dominant than any other. Consequently, we see from our simulation how quickly or slowly *The New Way Forward* worked. Conversely, we also see how earlier troop reductions (which happened for $t \in [31,39]$) may have been premature. This is not a firm result but only a probability-based one coming from the distribution over Markov-chains. More research would be needed to see exactly how the decision-loop and feed-forward inference interact, not to mention how that loop might affect feedback inference.

Decision-Making with a Progressive Extreme-case Network

These kinds of situations – or games – can be played with one of the networks trained on progressively extreme-case data too. Recall that for these networks the progression of unstable states is not as gradated as it was for the networks trained on monotonic extreme-cases. Nevertheless, we know that one Markov-chain (*g1*) was the most likely towards the latter stages of progressive instability. Consequently, our working hypothesis is that we are able to follow that Markov-chain in inference on real data to gauge instability of a given month.

Let us consider the $N = 16$ network trained on the progressively extreme-case data (see appendix C.9). Recall that, after training, this network's top-level node recognized *g1* in the latter periods of progressively unstable situations. Consequently, this led us to the conclusion that *g1* represents a state of the bottom-up evidence that is closer to instability than any of the other Markov-chains. So how would the decision-loop function for this computational SA?

Since the significance of other top-level Markov-chains is not clear, our only guide is the probability of *g1* at each month. Let us assume again that a decision-maker assesses monthly feed-forward inference on the $N = 16$ metrics. Except now the decision-maker tracks the probability of *g1* each month to gauge the instability – and implied stability – level. If we consider $t \in [0,8]$ (see Figure 79) then we see that *g1* is the least likely Markov-chain.



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
0	Group 7 1.677514	Group 2 1.586204	Group 0 1.526612	blank 1.489796	Group 4 1.465829	Group 6 1.01452	Group 5 0.955507	Group 3 0.954928	Group 1 0.000139
1	Group 0 1.224857	Group 4 1.035763	Group 7 1.022173	Group 2 1.007358	Group 5 0.979954	Group 3 0.976529	Group 6 0.885199	blank 0.690392	Group 1 0.001132
2	Group 0 1.691301	Group 7 1.656679	Group 4 1.442435	Group 2 1.35783	Group 5 0.97021	Group 3 0.952416	Group 6 0.70355	blank 0.517855	Group 1 9.472e-05
3	Group 0 1.277755	Group 4 1.251832	Group 2 1.210828	Group 7 1.117154	Group 5 0.984342	Group 3 0.963493	Group 6 0.954181	blank 0.611345	Group 1 0.001123
4	Group 0 1.873417	Group 2 1.762089	Group 4 1.599245	Group 7 1.47474	Group 5 1.313794	Group 3 1.271151	Group 6 1.159823	blank 0.661304	Group 1 6.167e-05
5	Group 0 1.517632	Group 5 1.460812	Group 3 1.359087	Group 2 1.318076	Group 4 1.279103	Group 6 1.183471	Group 7 0.982039	blank 0.487566	Group 1 0.000919
6	Group 5 1.536353	Group 0 1.534006	Group 3 1.489165	Group 2 1.457269	Group 6 1.412427	Group 4 1.135766	Group 7 0.964146	blank 0.485765	Group 1 0.001413
7	Group 3 1.650042	Group 5 1.639602	Group 0 1.637854	Group 6 1.614242	Group 2 1.25363	Group 4 1.107794	Group 7 0.909764	blank 0.578785	Group 1 0.001496
8	Group 5 1.659402	Group 0 1.649887	Group 3 1.640696	Group 6 1.459331	Group 2 1.098955	Group 4 0.951093	Group 7 0.775398	blank 0.464865	Group 1 0.000533
9	Group 5 1.763319	Group 0 1.749684	Group 3 1.715608	Group 6 1.680968	Group 2 1.433043	Group 4 1.324391	Group 7 1.076979	blank 0.640226	Group 1 0.000226
10	Group 0 1.615878	Group 5 1.603235	Group 3 1.543161	Group 6 1.434336	Group 2 0.96122	Group 4 0.878496	Group 7 0.748347	blank 0.448713	Group 1 0.001089
11	Group 5 0.81484	Group 0 0.798091	Group 3 0.694751	Group 6 0.617489	Group 2 0.358279	Group 4 0.236076	Group 1 0.207254	Group 7 0.124889	blank 0.018161
12	Group 0 1.1284	Group 5 1.080914	Group 3 0.939211	Group 2 0.934562	Group 4 0.815045	Group 6 0.750723	Group 7 0.530912	blank 0.096418	Group 1 0.013967
13	Group 0 1.660396	Group 5 1.596873	Group 3 1.434334	Group 2 1.260122	Group 4 1.203923	Group 6 1.178761	Group 7 0.909868	blank 0.378485	Group 1 0.002042
14	Group 0 1.608481	Group 5 1.561608	Group 3 1.372282	Group 6 1.095366	Group 2 1.036615	Group 4 0.998612	Group 7 0.761404	blank 0.361282	Group 1 0.005507

Figure 79: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [0,14]$

But we can track its movement to see something interesting about the SA formed by this network. At $t = 0$, the probability of *g1* is low (*0.001132*) and so the decision-maker would take this as an indication of low levels of instability. At $t = 1$, the probability of *g1* drops even further and so operations would continue as they have been to augment stability. Then at $t = 2$, the probability of *g1* increases to a value near its value at $t = 0$.

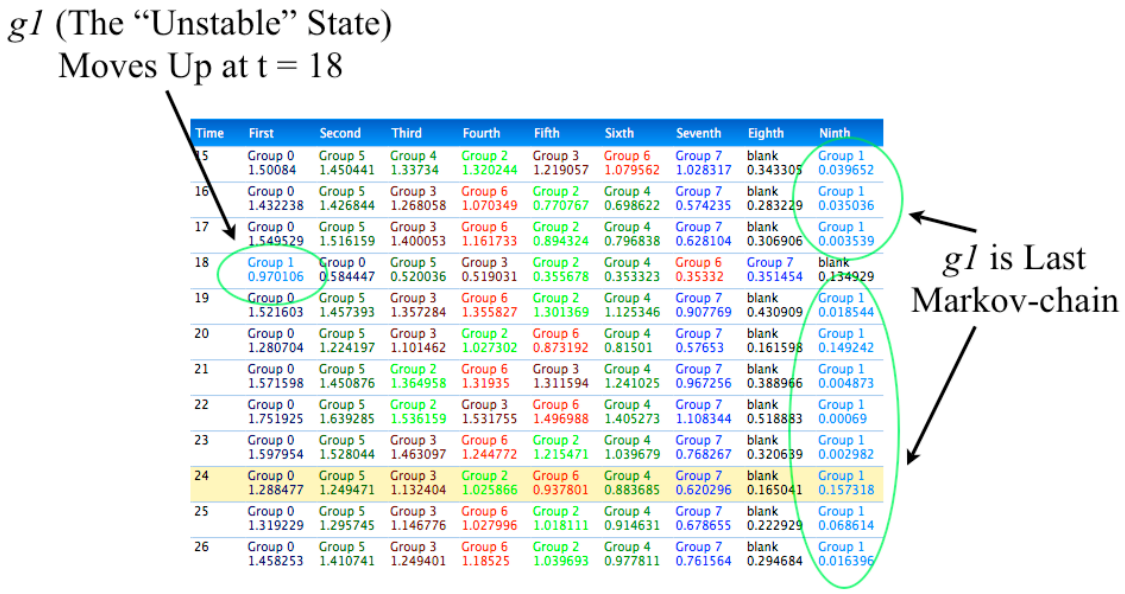
Then at $t = 3$ there is a drop again. Despite these fluctuations, a monthly evaluation of the probability of *g1* indicates that instability is relatively low. So the decisions in response to these probabilities might simply continue the *status quo*.

It is interesting to note though that this gradation to instability is not exactly what the monotonic extreme-case network showed. But for $t \in [0,8]$, the probability distribution of that network had indicated relatively low-number Markov-chains. Consequently, the qualitative recognition of these networks is similar during this time period.

Now let us consider the probability of *g1* for $t \in [9,14]$ (Figure 79). At $t = 9$, the probability of *g1* is still relatively low and *g1* is the least likely Markov-chain still. So a decision-maker would recognize this and continue operations as they have been. At $t = 10$, the probability of *g1* increases by an order of magnitude. There is no cause for considerable alarm though because it has had greater values at other times (e.g., $t = 1$) and it is still relatively low. So operations continue as they have been. But at $t = 11$, the probability of *g1* increases by two orders of magnitude and now *g1* is the seventh most likely Markov-chain. So perhaps the decision-maker responds and alters operations. Or, perhaps, he/she does not. We do not presume to know, but he/she would see at $t = 12$ that *g1* has returned to be the least likely Markov-chain. Its probability falls more and stays relatively low through $t = 14$. Consequently, actions can more-or-less maintain the *status quo* for $t \in [9,14]$.

Once again, although this is not exactly what was seen from the SA of the monotonically trained network, it is somewhat comparable. For instance, both networks recognized a substantial change in the state of the data at $t = 11$. Furthermore, both recognized a somewhat graceful falloff in recognition of unstable states through $t = 14$.

We can look at the decision-loop possibilities for $t \in [15,26]$. For this entire time period, *g1* is the least likely Markov-chain, except at $t = 18$ (Figure 80).

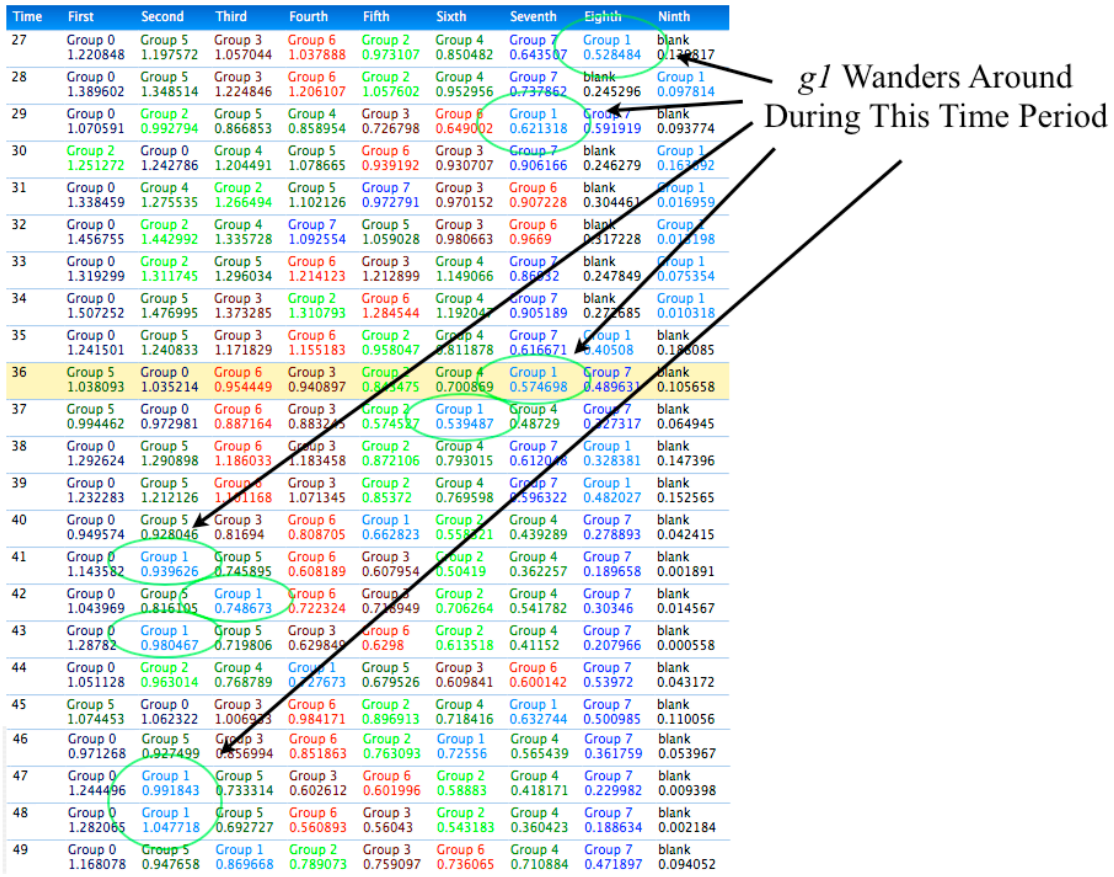


Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
15	Group 0 1.50084	Group 5 1.450441	Group 4 1.33734	Group 2 1.320244	Group 3 1.219057	Group 6 1.079562	Group 7 1.028317	blank 0.343305	Group 1 0.039652
16	Group 0 1.432238	Group 5 1.426844	Group 3 1.268058	Group 6 1.070349	Group 2 0.770767	Group 4 0.698622	Group 7 0.574235	blank 0.283229	Group 1 0.035036
17	Group 0 1.549529	Group 5 1.516159	Group 3 1.400053	Group 6 1.161733	Group 2 0.894324	Group 4 0.796838	Group 7 0.628104	blank 0.306906	Group 1 0.003539
18	Group 1 0.970106	Group 0 0.584447	Group 5 0.520036	Group 3 0.519031	Group 2 0.355678	Group 4 0.353323	Group 6 0.35332	Group 7 0.351454	blank 0.134929
19	Group 0 1.521603	Group 5 1.457393	Group 3 1.357284	Group 6 1.355827	Group 2 1.301369	Group 4 1.125346	Group 7 0.907769	blank 0.430909	Group 1 0.018544
20	Group 0 1.280704	Group 5 1.224197	Group 3 1.101462	Group 2 1.027302	Group 6 0.873192	Group 4 0.81501	Group 7 0.57653	blank 0.161598	Group 1 0.149242
21	Group 0 1.571598	Group 5 1.450876	Group 2 1.364958	Group 6 1.31935	Group 3 1.311594	Group 4 1.241025	Group 7 0.967256	blank 0.388966	Group 1 0.004873
22	Group 0 1.751925	Group 5 1.639285	Group 2 1.536159	Group 3 1.531755	Group 6 1.496988	Group 4 1.405273	Group 7 1.108344	blank 0.518883	Group 1 0.00069
23	Group 0 1.597954	Group 5 1.528044	Group 3 1.463097	Group 6 1.244772	Group 2 1.215471	Group 4 1.039679	Group 7 0.768267	blank 0.320639	Group 1 0.002982
24	Group 0 1.288477	Group 5 1.249471	Group 3 1.132404	Group 2 1.025866	Group 6 0.937801	Group 4 0.883685	Group 7 0.620296	blank 0.165041	Group 1 0.157318
25	Group 0 1.319229	Group 5 1.295745	Group 3 1.146776	Group 6 1.027996	Group 2 1.018111	Group 4 0.914631	Group 7 0.678655	blank 0.222929	Group 1 0.068614
26	Group 0 1.458253	Group 5 1.410741	Group 3 1.249401	Group 6 1.18525	Group 2 1.039693	Group 4 0.977811	Group 7 0.761564	blank 0.294684	Group 1 0.016396

Figure 80: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [15,26]$

For $t \in [15,17]$, the probability of *g1* is higher than it has been but it is still quite low. So the decision-maker could track this and maintain operations as they have been. But at $t = 18$, the huge jump in *g1* to be the most likely Markov-chain would warrant some concern. The decision-maker can read this though and from that knowledge make a decision with regards to stability. In the next time step, *g1* has dropped considerably. It only shows relatively elevated values at $t \in \{20,24\}$. As with other time steps, the decision-maker could use this information as an indication of an augmented instability level. If we look back to the inference of the monotonically trained network, then we see that these two time points are where the probability distribution shifted towards *g12, g13* and other higher-number Markov-chains. So once again we see some overlap in the SA of these two networks. So a decision-maker could use either one and make comparable conclusions about the stability level.

For $t \in [27,49]$ (see Figure 81), we can also simulate the decision-loop.

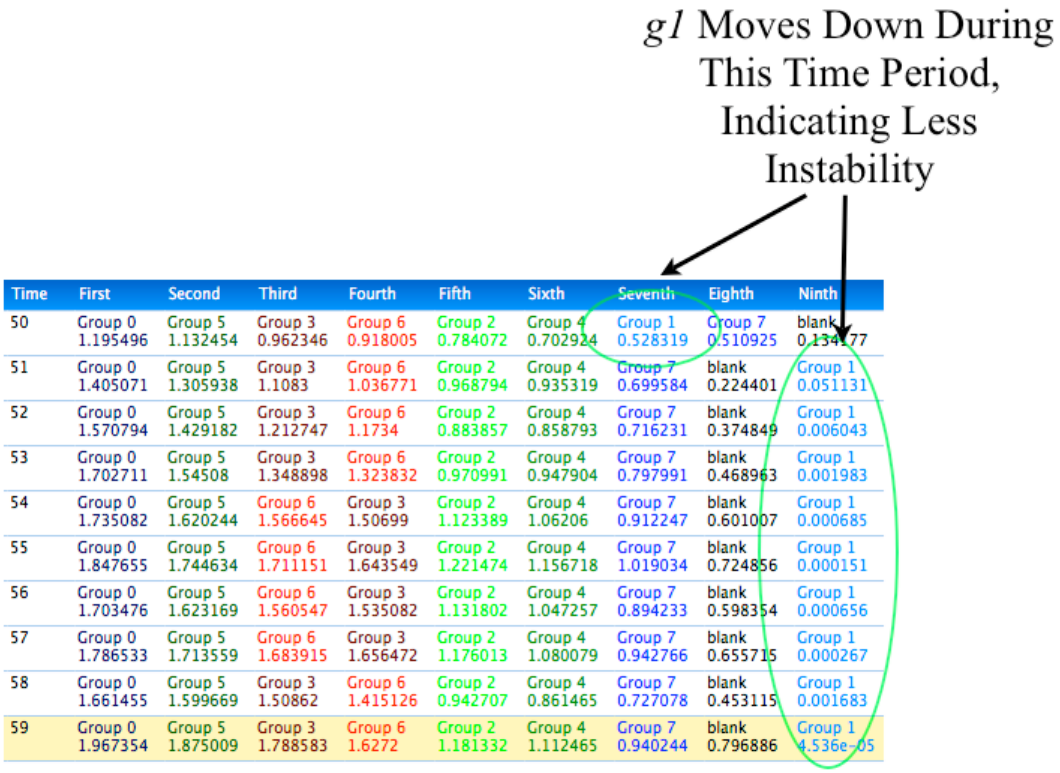


Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
27	Group 0 1.220848	Group 5 1.197572	Group 3 1.057044	Group 6 1.037888	Group 2 0.973107	Group 4 0.850482	Group 7 0.643507	Group 1 0.528484	blank 0.137817
28	Group 0 1.389602	Group 5 1.348514	Group 3 1.224846	Group 6 1.206107	Group 2 1.057602	Group 4 0.952956	Group 7 0.737862	blank 0.245296	Group 1 0.097814
29	Group 0 1.070591	Group 2 0.992794	Group 5 0.866853	Group 4 0.858954	Group 3 0.726798	Group 6 0.649002	Group 1 0.621318	Group 7 0.591919	blank 0.093774
30	Group 2 1.251272	Group 0 1.242786	Group 4 1.204491	Group 5 1.078665	Group 6 0.939192	Group 3 0.930707	Group 7 0.906166	blank 0.246279	Group 1 0.163892
31	Group 0 1.338459	Group 4 1.275535	Group 2 1.266494	Group 5 1.102126	Group 7 0.972791	Group 3 0.970152	Group 6 0.907228	blank 0.304461	Group 1 0.016959
32	Group 0 1.456755	Group 2 1.442992	Group 4 1.335728	Group 7 1.092554	Group 5 1.059028	Group 3 0.980663	Group 6 0.9669	blank 0.917228	Group 1 0.012198
33	Group 0 1.319299	Group 2 1.311745	Group 5 1.296034	Group 6 1.214123	Group 3 1.212899	Group 4 1.149066	Group 7 0.86032	blank 0.247849	Group 1 0.075354
34	Group 0 1.507252	Group 5 1.476995	Group 3 1.373285	Group 2 1.310793	Group 6 1.284544	Group 4 1.192047	Group 7 0.905189	blank 0.272585	Group 1 0.010318
35	Group 0 1.241501	Group 5 1.240833	Group 3 1.171829	Group 6 1.155183	Group 2 0.958047	Group 4 0.811878	Group 7 0.616671	Group 1 0.40508	blank 0.166085
36	Group 5 1.038093	Group 0 1.035214	Group 6 0.954449	Group 3 0.940897	Group 7 0.815475	Group 4 0.700869	Group 1 0.574698	Group 7 0.489631	blank 0.105658
37	Group 5 0.994462	Group 0 0.972981	Group 6 0.887164	Group 3 0.883245	Group 2 0.574517	Group 1 0.539487	Group 4 0.48729	Group 7 0.327317	blank 0.064945
38	Group 0 1.292624	Group 5 1.290898	Group 6 1.186033	Group 3 1.183458	Group 2 0.872106	Group 4 0.793015	Group 7 0.612018	Group 1 0.328381	blank 0.147396
39	Group 0 1.232283	Group 5 1.212126	Group 6 1.101168	Group 3 1.071345	Group 2 0.85372	Group 4 0.769598	Group 7 0.596322	Group 1 0.482027	blank 0.152565
40	Group 0 0.949574	Group 5 0.928046	Group 3 0.81694	Group 6 0.808705	Group 1 0.662823	Group 2 0.558621	Group 4 0.439289	Group 7 0.278893	blank 0.042415
41	Group 0 1.143582	Group 1 0.939626	Group 5 0.745895	Group 6 0.608189	Group 3 0.607954	Group 2 0.50419	Group 4 0.362257	Group 7 0.189658	blank 0.001891
42	Group 0 1.043969	Group 5 0.816105	Group 1 0.748673	Group 6 0.722324	Group 3 0.716949	Group 2 0.706264	Group 4 0.541782	Group 7 0.30346	blank 0.014567
43	Group 0 1.28782	Group 1 0.980467	Group 5 0.719806	Group 3 0.629846	Group 6 0.6298	Group 2 0.613518	Group 4 0.41152	Group 7 0.207966	blank 0.000558
44	Group 0 1.051128	Group 2 0.963014	Group 4 0.768789	Group 1 0.727673	Group 5 0.679526	Group 3 0.609841	Group 6 0.600142	Group 7 0.53972	blank 0.043172
45	Group 5 1.074453	Group 0 1.062322	Group 3 1.006933	Group 6 0.984171	Group 2 0.896913	Group 4 0.718416	Group 1 0.632744	Group 7 0.500985	blank 0.110056
46	Group 0 0.971268	Group 5 0.927499	Group 3 0.856994	Group 6 0.851863	Group 2 0.763093	Group 1 0.72556	Group 4 0.565439	Group 7 0.361759	blank 0.053967
47	Group 0 1.244496	Group 1 0.991843	Group 5 0.733314	Group 3 0.602612	Group 6 0.601996	Group 2 0.58883	Group 4 0.418171	Group 7 0.229982	blank 0.009398
48	Group 0 1.282065	Group 1 1.047718	Group 5 0.692727	Group 6 0.560893	Group 3 0.56043	Group 2 0.543183	Group 4 0.360423	Group 7 0.188634	blank 0.002184
49	Group 0 1.168078	Group 5 0.947658	Group 1 0.869668	Group 2 0.789073	Group 3 0.759097	Group 6 0.736065	Group 4 0.710884	Group 7 0.471897	blank 0.094052

Figure 81: Using Top-Level Node's Feed-forward Inference for Decision-Making, *t ∈ [27,49]*

For *t ∈ [27,49]*, we see *g1* wander around the probability distribution over Markov-chains. For instance, it is the seventh, sixth, fifth and fourth most likely Markov-chain at times, even becoming the second and third most likely for *t ∈ [41,43]*. This "wandering" is somewhat in line with what the monotonically trained network recognized. So a decision-maker would use the probability of *g1* as an indicator that instability comes and goes during this period of time. In fact, for *t ∈ {29, [35,49]}*, *g1* is not the least likely Markov-chain anymore, possibly indicating that some degree of instability exists in Iraq during these times. The correspondence of this SA to that of the monotonically trained network is not exact, but there is still some non-trivial correspondence.

Now, we look at $t \in [50,59]$. The decision-maker observes that *g1* is the seventh most likely Markov-chain (Figure 82).



Time	First	Second	Third	Fourth	Fifth	Sixth	Seventh	Eighth	Ninth
50	Group 0 1.195496	Group 5 1.132454	Group 3 0.962346	Group 6 0.918005	Group 2 0.784072	Group 4 0.702924	Group 1 0.528319	Group 7 0.510925	blank 0.134177
51	Group 0 1.405071	Group 5 1.305938	Group 3 1.1083	Group 6 1.036771	Group 2 0.968794	Group 4 0.935319	Group 7 0.699584	blank 0.224401	Group 1 0.051131
52	Group 0 1.570794	Group 5 1.429182	Group 3 1.212747	Group 6 1.1734	Group 2 0.883857	Group 4 0.858793	Group 7 0.716231	blank 0.374849	Group 1 0.006043
53	Group 0 1.702711	Group 5 1.54508	Group 3 1.348898	Group 6 1.323832	Group 2 0.970991	Group 4 0.947904	Group 7 0.797991	blank 0.468963	Group 1 0.001983
54	Group 0 1.735082	Group 5 1.620244	Group 6 1.566645	Group 3 1.50699	Group 2 1.123389	Group 4 1.06206	Group 7 0.912247	blank 0.601007	Group 1 0.000685
55	Group 0 1.847655	Group 5 1.744634	Group 6 1.711151	Group 3 1.643549	Group 2 1.221474	Group 4 1.156718	Group 7 1.019034	blank 0.724856	Group 1 0.000151
56	Group 0 1.703476	Group 5 1.623169	Group 6 1.560547	Group 3 1.535082	Group 2 1.131802	Group 4 1.047257	Group 7 0.894233	blank 0.598354	Group 1 0.000656
57	Group 0 1.786533	Group 5 1.713559	Group 6 1.683915	Group 3 1.656472	Group 2 1.176013	Group 4 1.080079	Group 7 0.942766	blank 0.655715	Group 1 0.000267
58	Group 0 1.661455	Group 5 1.599669	Group 3 1.50862	Group 6 1.415126	Group 2 0.942707	Group 4 0.861465	Group 7 0.727078	blank 0.453115	Group 1 0.001683
59	Group 0 1.967354	Group 5 1.875009	Group 3 1.788583	Group 6 1.6272	Group 2 1.181332	Group 4 1.112465	Group 7 0.940244	blank 0.796886	Group 1 4.536e-05

Figure 82: Using Top-Level Node's Feed-forward Inference for Decision-Making, $t \in [50,59]$

Then it drops to the least likely Markov-chain and so operations would continue as they have been, since this change in ranking indicates progress towards the goal of stability. Each month after this, the probability of *g1* either drops or stays low, indicating for the decision-maker an ability to recognize the consequences of his/her actions.

Once again, the comparison to the monotonically trained network is not perfect. For instance, the rise and fall in probability of *g1* does not exactly match the movement of the probability distribution over the top-level node's Markov-chains. But the general trend is still the same: instability diminishes each month from $t = 50$ to $t = 59$. In the monotonically trained network, diminishing instability is indicated with the probability distribution over all groups. For this network, it is indicated with the probability of *g1*.

For either case, a level of SA has been created computationally that would allow a decision-maker to adjust operations in response to a universal metric on stability/instability.

For both networks, we are confident in the top-level node's conclusions because of the training strategies done here. For both, we have given an HTM network boundaries it can use to gauge progress of a given situation. So the month-by-month inference of actual data is done with respect to a scale of stability. To be exact, in our implementation, it has been a scale of instability. But due to the monotonic utility of stability/instability one implies the other.

Visualizing Both Extreme-case Networks' Decision-Making Support

We can further facilitate the decision-making utility of these networks with visualization of the top-level node's output. Recall that when the network gauges the situation, it does so by classification and this is the language of the decision-maker. Human factors research has showed us that this method of information processing – categorization – is very much how actual decision-makers operate. The final step to communicating that information is with a visual aid. For each of these networks, the categorization is done via monitoring a universal metric or set of metrics. For instance, for the monotonically trained network, we followed the entire probability distribution over the top-level node's Markov-chains. For the progressively trained network, we followed the probability of *g1*. Both ways of categorizing the situation can be visualized, but each in a different way.

The monotonically trained network provides decision support via the horizontal location of the top-level node's probability distribution. We can visualize the top-level node's output for $t \in [9,12]$ in Figure 83. From the figure, we can see how a decision-maker needs only to monitor the position of the Markov-chain probability distribution at

each time point to gauge instability. In Figure 83, the highest ten values from this distribution for $t \in [9,12]$ have been plotted.

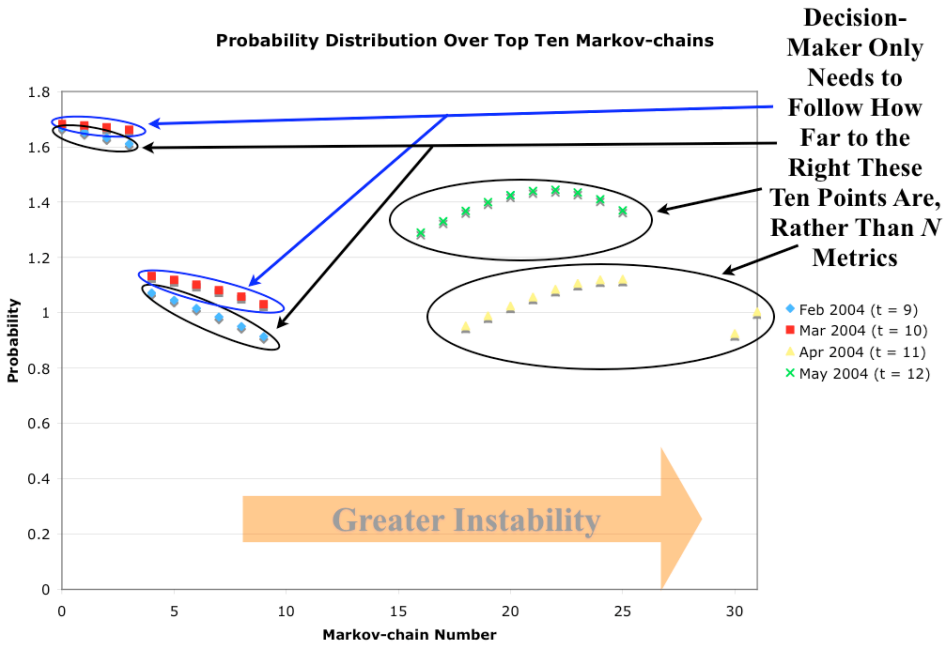


Figure 83: Visualized Output from Top-Level Node for Monotonically Trained Network, $t \in [9,12]$

Probability distributions that are further to the right (e.g., April and May 2004, i.e., $t \in [11,12]$) reflect more instability in the bottom-up evidence than those further to the left (e.g., February and March 2004, i.e., $t \in [9,10]$). Other time periods can be visualized in a similar manner, so that the decision-maker needs only to follow the left-right movement of the distribution each month. With this nontrivial categorization step already achieved, the decision-maker can then proceed to a suitable course of action.

A similar argument can be made for the progressively trained network, but the decision-maker would follow a different metric. Since that network identified only one state as being conclusively unstable ($g1$), the decision-maker would follow the probability of that Markov-chain, given the bottom-up evidence each month. So the data

of Figure 79 through Figure 82 can be visualized and used that way by the decision-maker, as shown in Figure 84.

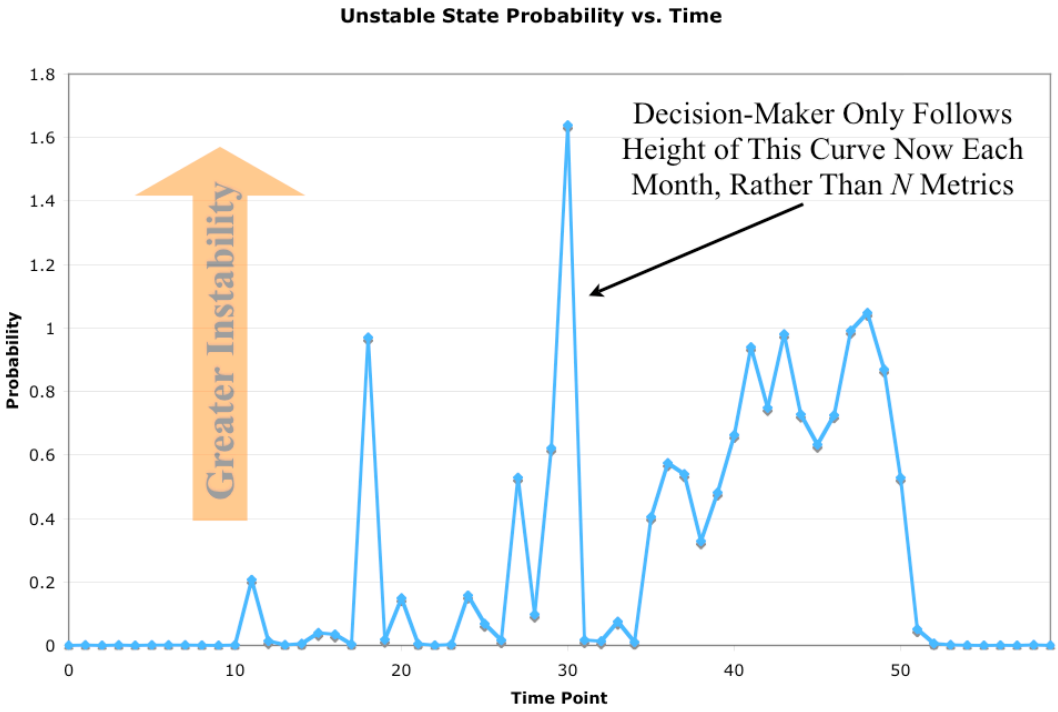


Figure 84: Visualized Output from Top-Level Node for Progressively Trained Network, *t ∈ [0,59]*

Here, we see that the bottom-up evidence of each month (i.e., time point) is categorized by a probability value of how unstable Iraq is. As with the monotonically trained network, having achieved this categorization step, the decision-maker can then proceed to suitable action.

Consequently, the output of both networks speaks the language of the decision-maker. He/she wants to know and to see how stable or unstable the situation is each month. **The output of these networks can visually communicate to the decision-maker how unstable Iraq is each month.** From this assessment, he/she can decide on a course of action. We can only simulate this decision-loop here by unknown or hypothetical actions but the assessment each month is the starting point for these actions. As Endsley reminds us [7], "There is considerable evidence that a person's manner of

characterizing a situation will determine the decision process chosen to solve a problem." So we see here that the decision process about the Iraq War might have been different if a decision-maker had access to such a synthesis of information.

In summary, in the previous chapter, we fused results from human factors with techniques from machine learning to form a useful schema about a given complex system. In this chapter, we have shown how this schema could be of use in decision-making. If we were to evaluate the full efficacy of it though, we would then require control over these decisions. But since such analysis is considerably beyond the scope of this research, we leave it here knowing that a new possibility exists for decision-making support. Instead, we turn now to assess what results we have contributed to complex system analysis.

Implications for Complex System Analysis

In terms of complex system analysis, our primary result is that the goal of the analysis shapes the schema formation. In using HTM to learn and to recognize features of a complex system, we have developed a new technique for complex system analysis. In some respects, our result is not too surprising because we have used insights from complex systems research as our guide. For instance, we saw the importance of coarse graining in the literature review. This choice of observational level of detail is somewhat linked to the goal of any analysis. Consequently, this result is somewhat of a confirmation of earlier work, but by incorporating the decision-loop we have explicitly tied analysis to goals of operation. This does not mean that there is no way to analyze a complex system when there is no goal to be satisfied by operating on it. But, when there is a goal, this focuses attention to the elements of the system that are relevant to the decision-maker. Consequently, it allows schema formation from a finite-dimensional $V_{properties}$ vector space, resulting in a degree of SA on that complex system. If anything, the primary result here is that complex system analysis can be done with an approach

based on SA. Furthermore, the implementation of SA we have done is computational and so it is penetrable to intermediate levels of inquiry, whereas human SA is less so.

Summary of Results

We have seen how the previous chapter's analysis leads us to a new way of analyzing complex systems. In this chapter, we have used this analysis for pseudo-decision-loop scenarios. Two examples have been probed here and they have not been equal in their level of complexity. The shockwaves context could be coarse grained at a level that satisfies properties of simpler physical systems. From that data, we develop a knowledge base about flow patterns and shocks. But then in a decision-making environment, we are able to use this knowledge base to respond to significant changes in the flow. The exact hardware and software for doing this has not been discussed here, but the mechanism is quite clear. The decision-maker (or software) monitors the value of λ_t of the top-level node and enacts a decision (via hardware) to meet goals. What these goals are depends on the situation. Are we trying to avoid shocks? Are we trying to operate the engine within a range of properties? These questions would guide the loop between software and hardware.

Similarly, the knowledge base we form on the Iraq context can also be utilized in a decision-loop. But the exact "hardware" and "software" is different. In this context, the "hardware" is the U.S. military. Consequently, the "software" must interface with this instrument to enact decisions. Avoiding these details though, we see from our simulation of a decision-maker reading λ_t monthly that the network could be a key piece to the decision-loop. Are we trying to augment stability? Based on the answers to this question, we can focus our attention to output of the network that answers our progress in attaining this end state. Consequently, decisions based off of this evidence-based approach are possible.

While there are some shortcomings here and there, the overall approach seems to work for at least two of the networks probed in this research. We can deduce that similar results would follow for using other networks in this way. But when assessing these results, we have to recall an earlier insight from the literature review [33]: "The success of any Situation Awareness system depends upon understandable Measures of Performance (MOP) and Measures of Effectiveness (MOE). These measures must include quantitative and qualitative characterizations and be directly tied to the mission of the system in question." Our quantitative characterization has been the setting of extreme-case bounds for training and the resulting top-level node Markov-chains generated from learning about such data. This has translated into the ability to notice trends in instability recognition. Our qualitative characterization has been to notice the correspondences between general recognition trends of different networks. These correspondences were seen during comparable time periods of the actual data on the Iraq context. Consequently, we may have contributed a new computational approach to SA by using HTM networks.

Finally, we have noticed a result that applies to complex systems research. Specifically, we have analyzed a complex system by using an approach to information processing derived from SA theory. But our extension of this research has brought it into a computational domain that allows its intermediate results to be readily probed. Consequently, this is a potential insight into complex system analysis.

In light of these results, two re-examinations will be of concern in the next chapter. First, we shall re-examine our implementation in light of our hypothesis taxonomy. Second, we will re-consider the baseline examples of computational SA discussed in the literature review. Finally, after this comparison, we will then assess our results from a more general perspective, putting them in the context of complex systems research.

CHAPTER 7

CONCLUDING REMARKS

Our experiments, analysis and results lead us now to reconsider our original hypotheses. In doing so, we are able to gauge what has been accomplished here and how it fits into the context of the state-of-the-art. First, we proceed to revisit the hypothesis taxonomy proposed in the third chapter. Along the way, we will see how our results either support or discredit each hypothesis. After doing so, we will then see what contributions to the state-of-the-art have been made here.

Revisit Hypothesis Taxonomy

The hypothesis taxonomy responded to the research questions and so we see now how our experiments compare to these assertions. We will re-state and then discuss observations relevant to each of them. We begin by examining sub-hypotheses related to the first hypothesis (H A). The first sub-hypothesis of H A is as follows:

a. For a given context, the available data can be leveraged simultaneously when it is condensed into a schema that is suitable to the specified goals.

We examined two contexts in our implementation. For each of them, data was used as a basis to form a schema for particular goals. Furthermore, all of the data was considered at once at each time point. In other words, every piece of data available at a given time point was considered simultaneously. For instance, in the shockwaves context, the simulated data on high-speed flow provided four properties at each time point. From the time-series of these four properties, the learning algorithms of an HTM network condensed the data into a schema. If the goal was flow pattern recognition (e.g., is it hotter, more pressurized, etc.?) then an unsupervised network was the foundation of the

schema. If the goal was shock recognition then a supervised network was the foundation. From these schemata, new evidence at each time point could be presented to the networks, allowing them to propagate evidence through its levels and produce λ_t. One piece of data was never favored over any other in the recognition of flow patterns or shocks. Rather, all pieces of data were always considered simultaneously as one piece of evidence.

A similar result was seen for the Iraq context. The time-series of $N \geq 8$ metrics from fictitious extreme-case data allowed us to form schemata for tracking instability gradations. After training, the networks were able to consider all pieces of evidence concurrently and then output λ_t as an assessment. Some networks gave more consistent results than others. For example, the unsupervised ones trained on monotonic/progressive extreme-case data seemed to recognize gradations of instability. These schemata were tailored specifically to the goal of stability because the data used to train them was selected this way.

The second sub-hypothesis concerns the hierarchical manner by which we have chosen to condense observed data:

b. A hierarchical condensation of the data is possible, but the significance of it remains to be seen.

In the literature review, we introduced HTM as a means to hierarchically condense data. But it was not until we went through the details of learning and inference in nodes that we saw the way spatial and temporal information is compressed. We were then in a position to take these results and extend them to the shockwaves and Iraq contexts. Specifically, the HTM networks used to analyze high-speed flow allowed us to categorize certain types of flows and shock conditions. Also, the hierarchical condensation of data on the Iraq context allowed us to categorize actual data with which it could then produce

a stability assessment. Furthermore, both the shockwaves and Iraq context networks can produce output in the real-time of each context. For shockwaves, this is on the order of $\sim 10^{-3}$ seconds. For Iraq, it is on the order of a month. This allows decision-making then to be based on such output.

The third sub-hypothesis focuses on the importance of time in the datasets from which schemata are formed:

c. For a specified context and goals, a temporally structured dataset can be prepared and analyzed to extract high-level states evident in the data, although the meaning of these states remains to be tested.

For both contexts, temporally structured datasets were the key to schema formation. For the shockwaves context, we simulated flow evolving in time whereby supersonic bodies served as factors affecting the flow properties. From this temporally structured dataset, the unsupervised network learned about flow patterns evident in the data. With the addition of supervision, both the $N = 2$ and $N = 4$ networks were able to extract information on shock formation. In both cases, the role of time has been important because it is a requirement for the generalization of HTM to other applications. Necessarily, this constrained us to prepare temporally structured datasets. This preparation amounted to Level 1 SA and their analysis amounted to Level 2 SA. For the shockwaves context, Level 1 SA included generating data from a model and then preparing it into format suitable for HTM learning. Then Level 2 SA included the training of this data, and the inference on this and other data. From training, the top-level node learned coincidence patterns (C) and Markov-chains (G), i.e., the high-level states found in the training data. A similar argument applies for the Iraq context.

The meaning of the top-level Markov-chains has been discussed throughout. For instance, in the Iraq context, we trained the network on progressive or monotonic

instability/stability. Consequently, the meaning of the top-level Markov-chains should have been clear. It turned out that only the meaning of Markov-chains for unstable states has been clear. Recall that this is because most of the metrics are inversely proportional to stability and so in extreme-case stable states they become too small to form non-zero *C* and *G*. Similarly, in the shockwaves context, depending on the presence or lack of supervision, the meaning of the top-level Markov-chains has been shown to correspond strongly with the flow patterns and shocks. Once again, the correspondence is not perfectly synchronous, but it is too frequent to ignore.

The fourth and final sub-hypothesis of H A concerns the SA of stability assessment:

d. To analyze an implementation of SA in the complex task of stability assessment, at least two approaches are needed: expert validation and extreme-case bounding.

As the fifth chapter showed, the extreme-case bounding has been an invaluable insight into the significance of our results. It has provided a reference point for us to gauge recognition when validation is not an easy matter. This allowed inference on real data to be judged with respect to some quantitative background. We also attempted the use of some expert validation from the DoD. But having used the extreme-case bounding so effectively both to prove and disprove inference results, the expert validation fell short in terms of guidance. As we saw from comparisons with DoD literature, the qualitative reports could not give us clear categories of stability or instability, though they frequently referred to a superset of the data we have considered. We could only gather from the DoD reports that our results are not wrong. We could not show with these reports that they were right. This conundrum is at the heart of why it is difficult to describe the stability of Iraq during 2003-2008. Furthermore, this extends to the difficulty of

analyzing any other NCW scenario. In light of this problem, we have found more consistent validation when extreme-cases are used as a reference point.

These four sub-hypotheses led up to one of our two primary hypotheses. Consequently, the observations pertinent to them are pertinent to it. This hypothesis was:

A. SA of decision-makers in NCW contexts possibly can be improved with computational aides that exploit the hierarchical and temporal structure of the data relevant to a given context and prescribed goal.

The Iraq context has been meant as an example NCW scenario. With a defined goal of stability, we were then able to find data pertinent to that goal. Using hierarchical and temporal condensation of synthetic data, we developed computational SA on the Iraq context. Some implementations worked better than others. In other words, some networks recognized certain time points as instable, while other recognized either the same or different time points in this way. But the end result from these networks is an ability to categorize the stability state based on actual evidence. Consequently, recalling the human factors research pointing to the importance of categorization, the SA of decision-makers might be enhanced with such an approach to situation analysis. This was then showed in the previous chapter, using two networks for simulated decision loops.

We can also reconsider the sub-hypotheses of the second primary hypothesis (H B). The first sub-hypothesis concerned the specific use of HTM as a means to fuse data and extract meaning from it:

a. Specifying a context and a set of goals, an approach based on Hierarchical Temporal Memory can balance the need to fuse data with that of extracting useful meaning from it.

Starting from the unsupervised version of the River (Waves) network, through the shockwaves and Iraq networks, we have seen how HTM can fuse data. The question of its meaning has been probed in detail by analysis of inference output (λ_t). For instance, we have seen how networks used in the shockwaves and Iraq contexts have fused N metrics' time-evolution into coincidence patterns (C) and Markov-chains (G) at all levels of the network. The top-level C and G is then what received the most analysis in terms of meaning because this node covered the entire receptive field of the context in question. Furthermore, it has been important to balance HTM nodes' abilities to fuse data with the significance of that fusion. For instance, we saw that three levels were not needed for the most accurate shockwaves networks. Instead, only two were needed to gauge flow patterns and recognize shocks. In contrast, no substantial modifications were made to the Iraq context networks because the top-level node's inference seemed to produce significant assessments of stability.

The second sub-hypothesis concerns the role of goals in assessment. Specifically, this hypothesis pinpoints how to compare alternate implementations of SA in the same context:

b. In assessment tasks of the same context, the goals determine the basis for comparison because these in turn determine the relevant observables.

For each context, we were able to compare alternative implementations because the data did not change. Even when the number of metrics was reduced, we still formed SA from relevant observables because they were information relevant to our goals. For instance, the various permutations of ordering the data in the sensors for the Iraq context could all be compared to each other because of the common goal of stability assessment. No matter what permutation was used, the goal remained the same and so did the metrics that were used. Our initial expectation had been that there would be no substantial changes in

the stability assessment. As we saw though, the ordering of the data did affect how the network learned to recognize features of stability and instability.

After reconsidering this initial expectation, it was actually an understandable result if we think of it analogically with the Pictures Problem. George's goal in that problem was to recognize different objects. But if we randomly permuted the order of the 1,024 pixels of each image, then there would be a different SA formed about this context. Each of the pixels would be just as relevant to the goal of object recognition. But assessment tasks in that context would necessarily produce different results because each image would be a jumbled mess. The network might still learn some patterns in the images, but they would likely not correspond to the objects we recognize with relative ease.

The third sub-hypothesis concerns the connection between the machine learning implementation and the formed SA:

c. For a given context, different machine learning algorithms – and even different implementations of the same algorithm – will likely create different SA in light of prescribed goals, but some techniques will yield more significant results than others.

We saw throughout the fifth chapter how different network configurations, ordering of the metrics, number of metrics, etc., led to different top-level coincidence patterns and Markov-chains. Given that the data in these implementations was relevant to SA formation (i.e. Level 1 SA), the top-level patterns and Markov-chains have been shown to correspond significantly to Level 2 SA. As we saw, certain configurations or training procedures led to more successful results than others. For instance, as we reduced the number of metrics in each context, the significance of the stability assessment and flow pattern recognition fell off.

Also, with regards to other machine learning algorithms, we touched briefly on the potential use of Euclidean distance as a means to handle the recognition of a shock. Specifically, this could either be done with a straightforward calculation serving as the categorization mechanism. Alternatively, some kind of *a priori* category data could also map the small changes in vectors to the same flow pattern, the mid-size changes to the presence of a shock, and the large changes to equilibrium recovery. Such an SA of the shockwaves problem might even outperform the $N = 4$ log-transformed network because of its greater flexibility in classifying flow patterns outside of its experience. Consequently, the actual way the SA is implemented strongly correlates with its abilities, even when the goals of different instantiations are the same.

We can combine the observations relevant to these sub-hypotheses and now address the second primary hypothesis:

B. It is likely that NCW SA can be formed quickly and maintained usefully through schema formation based on Hierarchical Temporal Memory.

The breadth of implementations of SA concerning the Iraq context should indicate the rapidity with which it can be formed. Specifically, this has been done with pertinent data when HTM is used for schema formation. Credit is also due to the Vitamin D Toolkit [129], which has provided a GUI that bypasses a possibly frustrating learning curve with Python, C++ or the Application Programmer Interface (API). Once a training dataset was in hand, it has been possible to form a schema directly from it. Furthermore, evaluation and re-evaluation of inference results has allowed us to demonstrate how this schema can be maintained or upgraded. This necessitates human-in-the-loop interaction via Level 1 SA, although the workings are there to possibly automate this feedback loop. But this possibility is not explored in this research. We only note it for possible future exploration.

Finally, the utility of some schemata has been seen with simulated decision-making problems for both the shockwaves and Iraq contexts.

In summary, each of the hypotheses and sub-hypotheses has been addressed in the course of executing our research plan. In doing so, we have been able to investigate a computational approach to the situational awareness of a complex system. By focusing on the shockwaves context (a somewhat simplified system), and the Iraq context (a more complex system), we have been able to demonstrate this approach and to see how it extends from analysis on simple physical systems. The common thread running through the analysis of either system has been a focus on the observable data that describes it. Finally, machine learning based on HTM has allowed us to implement this SA computationally. Having completed our research plan, let us now assess what contributions have been made to the state-of-the-art.

Contributions to the State-of-the-Art

The starting points for this research came from complex systems, human factors, machine learning and other related disciplines. Having gone through the implementation analysis in the fifth chapter and results in the sixth, we can summarize the contributions we have made, if any, to the state-of-the-art of these disciplines. We begin with a review of the benchmark cases discussed in the literature review, so that we can see how our implementation of SA compares to theirs.

Comparing HTM-based SA with Benchmark Cases

The literature survey identified three notable cases in which computational SA was attempted. To review, the first one considered a computational SA framework for decision-making in the context of the build up to the first Gulf War [33]. The second one considered a neural-network-based approach to sensory data fusion for force protection purposes [42]. While these first two cases are somewhat related to our example of NCW

SA, the third one considered strategic decision-making from a more general perspective. This third case looked at benchmark problems from that field [45]. Having seen the pros and cons to our Iraq context implementation with SA, we can now return to these cases to do some comparative analysis. Specifically, we can see how our approach with HTM has either advanced beyond these methods or remained behind them in utility. So we proceed to do this comparison now with the first case.

Case #1: A NCW Framework for Situation Awareness

This case is the least technically detailed of the three. It proposed a framework for perceiving and interpreting data related to a given scenario [33]. The chosen scenario is the Iraqi incursion to Kuwait that started the first Gulf War. The authors' goal in the analysis is to do something similar to what is depicted in Figure 16.

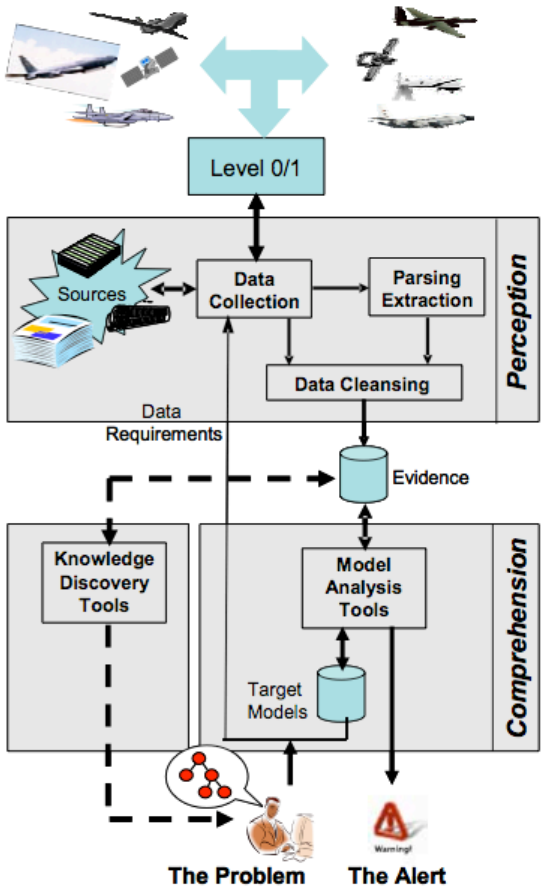


Figure 85: Situational Awareness Framework for NCW [33]

The breakdown of SA shown in this figure follows closely from Endsley's research. For instance, the "Perception" box is comparable to Level 1 SA and the "Comprehension" box is comparable to Level 2 SA and some of Level 3 SA.

If we recall the work done to implement SA with HTM networks then we can do some comparisons with these authors' work. The information flow (Figure 7) is in fact another version of Figure 85. But the authors implement their flow with an XML-based hierarchical decomposition of related metrics determined *a priori* (see Figure 6). Rather, our approach only assumes what data is worth perceiving. In fact, Level 1 SA completely takes over this function, assuming what "sources" (indicated in Figure 85) are useful. Then our implementation of Level 2 SA – comparable to their "Knowledge Discovery Tools" – proceeds from the training evidence to form the coincidence patterns and Markov-chains of each node. We do not specify connections between the metrics *a priori* in as detailed a fashion as these authors do. Our only specification of connections between the metrics had been to place related metrics within the same receptive field. For instance, four politico-economic metrics are fed into one node of the bottom level in the non-randomly-permuted ordering of metrics. Furthermore, we saw that the monotonic extreme-case networks are not as susceptible to this specification as the progressively extreme-case ones. But there is no *a priori* mapping done in our HTM implementation between context metrics. This represents a fundamental difference from the approach taken by these authors.

Case #2: A NCW Decision Support Implementation

Of the three cases, this one is the most similar to what we have done. Brannon et al. [38] attempt the mapping between raw sensor data and threat level for a potentially hostile environment. As with the previous case, there are similarities and differences between our method and theirs. This case though had the most effect in terms of how we analyzed our results, so a more step-by-step comparison to our implementation will be

done here. Specifically, we consider the differing choices about when to fuse the data and how this affects the situation assessment that follows. Also, we compare the interface each approach proposes for the decision-maker to engage the data.

Comparison of Fusion Approaches

The data from which computational SA is formed in Brannon et al. is relatively varied and raw in comparison to the data we used for the Iraq context. This difference in data influences the different fusion approaches employed in both implementations. For instance, recall that Brannon et al. used binary, categorical and real-valued data. For our coarse graining of the Iraq context, all data has been real-valued. Specifically, the metrics of the $N = 16\ V_{properties}$ (see appendices C.2 or C.3) are either integers or rational numbers. But for Brannon and colleagues' coarse graining of the force protection context, they have more varied inputs. For instance, binary sensors in the road indicate the presence of a vehicle; acoustic and seismic sensors indicate various details about the vehicle.

Having such differences between the data, there are considerable differences between the methods of executing situation assessment. Let us begin with Brannon and colleagues' implementation. They fuse the raw data into metrics indicating vehicle heading and speed, as well as others. Then these higher-level metrics, derived from the fusion of raw data, are passed through an *a priori* categorization of how threatening the value is. For instance, the vehicle speeds are partitioned across eleven classes. Then each range of vehicle speeds is mapped to a threat level for that metric. The same thing is done for vehicle heading, vehicle type, wind speed, etc. Each of these threat levels is then combined into a weighted sum. Consequently, this weighted sum gives the threat level as a function of bottom-up evidence. There is also room for reinforcement learning, which has not been discussed here. But in terms of unsupervised learning, this is about as close as their implementation comes to ours. Even without considering the reinforcement

learning, supervision acts directly on the values of each metric by categorizing them into threat levels.

We can contrast this now with our approach in the Iraq context. We started from data on various aspects to the stability of Iraq. Then we isolated those metrics for which a consistent time-series is available. We realized that training the network on fictitious extreme-cases allowed it to create a scale for categorizing instability and stability. Our only *a priori* instruction to the network has been in the selection of training data. In other words, we let the network learn the implicit information embodied in the extreme-cases. Some implementations worked better than others and generally instability was easier to learn than stability. But this allowed the network to form a schema for recognizing gradations in instability. With this knowledge base, we were then in a position to look at real data as it unfolds each month and to categorize the level of instability. There was no need to map the fused data to *a priori* categories of instability because the training process had done this. Furthermore, we showed earlier how a decision-maker might use this information in a sort of operation control loop.

Fundamentally, our fusion approach differs from Brannon et al. because of the differences in the data. The goals of each SA are similar though. They both aim at extracting information on threat level from data. In our implementation, the goal has been to extract stability information from data. Because of the monotonic extreme-case training, we have been able to do so with a comparable end result, i.e., an instability level.

Interface Between Data and Decision-Maker

There are some key differences to note between how each implementation interfaces with the decision-maker. The Brannon et al. implementation presents the results to the decision-maker in a GUI. In this interface, the decision-maker can see the raw evidence and a suggested threat level produced by this data. A great boon to this

interface is that it provides a channel for the user to provide reinforcement learning if the unsupervised learning is wildly errant. In fact, this interface is crucial to the neural networks running behind it and so it makes sense for this to be inviting for the user.

In our implementation, the unsupervised nature of the learning/inference determines the propriety of our interface. In our analysis, the Vitamin D Toolkit has been indispensable because it allows the user to visually see the inference output that serves as the indication of stability level. But there is no *en vivo* tuning of the learning as in the interface of Brannon et al. This is perhaps a drawback of our interface and HTM in general. But it is simple enough to incorporate the new data into a previously used data set and retrain the HTM network. The turn-around time is obviously short, since we analyzed dozens of networks and were able to make adjustments along the way.

In summary, the difference in the interfaces is a function of the difference in the learning algorithms. Brannon et al. allow for heavy supervision and they provide a channel for the user to provide it. For the Iraq context, we do not provide supervision at all. Once the network has trained, we provide the user with a way to load data that he/she wishes to categorize. Then the top-level node's probability distribution over Markov-chains provides the categorization. If the training dataset covers a suitable range of states (e.g., extreme-cases) then the actual state can be classified in relation to this scale.

Case #3: Strategic Decision-Making Support

As we saw in the literature review, the third case approaches decision-making from the perspective of benchmark problems [45]. Unfortunately, these benchmark problems have no clear relation to examples in which NCW SA is needed. Furthermore, Kohl and Miikkulainen assume in this study that the states of a given system are known and that the problem is to match them to goal-oriented actions. The goal then depends on the context in which the matching occurs. The authors then modify an evolved neural network to match the states to the actions.

But our analysis focused on characterizing the states Kohl and Miikkulainen assume to have as inputs. So it might be interesting to see how their approach could build on top of ours. This would require having a set of available actions a decision-maker could implement on the Iraq context. We see though that the comparison between our approach and that of Kohl and Miikkulainen is not direct. They assume our end result and so it is not easy to compare implementation or results.

Summary of Baseline Comparisons

There are pros and cons of each of the three approaches discussed above. Of course, the same is true of our approach. Also, since the contexts in which each are applied differ, it is not easy to compare them due to the substantial effects of different coarse graining. Nevertheless, similarities between the approaches can be seen. Furthermore, the work of Brannon et al. has provided somewhat of a procedure for how to present results. Also, the first case gave some idea about how Endsley's work on SA could be mapped to a computational implementation. Finally, although not discussed in-depth here, the last case gave a substantive background on the mathematical difficulties of extracting information from a complex system. This insight leads us to our next topic: the impact on complex system analysis. Specifically, now that we have re-considered and compared to the literature on computational SA, what can our implementation bring back to complex systems research?

SA Formation and Complex System Analysis

There are many issues from complex systems research that have been addressed in this research. The first concerns our proposal and implementation of a way to form a schema about a complex system. In doing so, we have not isolated any particular element of the system. Rather, we have studied it from a perspective that considers all elements on equal footing. This is the second issue we have addressed. Finally, we have extended

the line of thought explored in the literature review on the role of biological systems in aiding schema formation. This has explicitly been done with HTM as a rudimentary instantiation of cortical circuitry. We will now consider each of these results in more detail.

Our Approach to Schema Formation/Maintenance

We have synthesized research on situational awareness (SA) with HTM networks to propose and to implement computational SA. This has been done with a dynamic information flow (Figure 87) that allows for constant improvement of the formed schema. The two contexts explored in this work have allowed us to demonstrate this. One of these contexts was more complex than the other. But the technique has generally been the same. We have sought a way to determine what the question mark is in Figure 86. For the shockwaves context, this question mark represented the training and testing of HTM networks. For the Iraq context, this question mark represented system dynamics models to create extreme-cases and then the training/testing of HTM networks. The end result has been a degree of SA on each of these contexts.

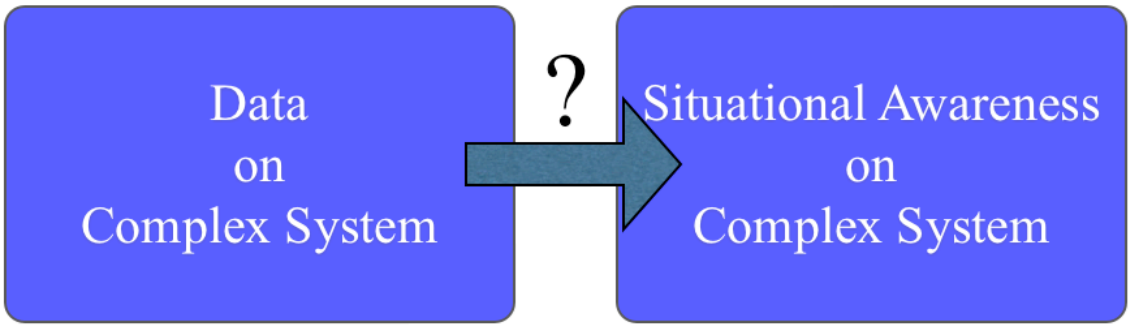


Figure 86: General Question Addressed with Research Plan

Although Figure 86 shows the process as being unidirectional, the information flow we have used throughout reminds us that it is not.

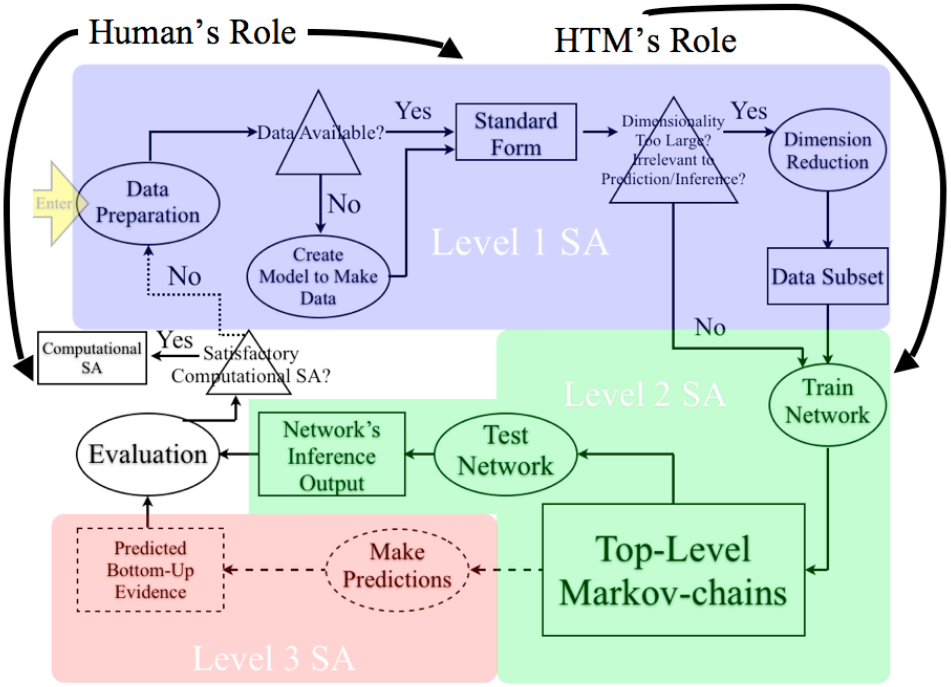


Figure 87: Modified Information Flow for HTM-based SA

Instead, the information flow requires evaluation and re-evaluation of the formed SA in light of goals. This flow of observation, training, testing, and evaluation, bears strong resemblance to Gell-Mann's description earlier about how a complex system works (Figure 88).

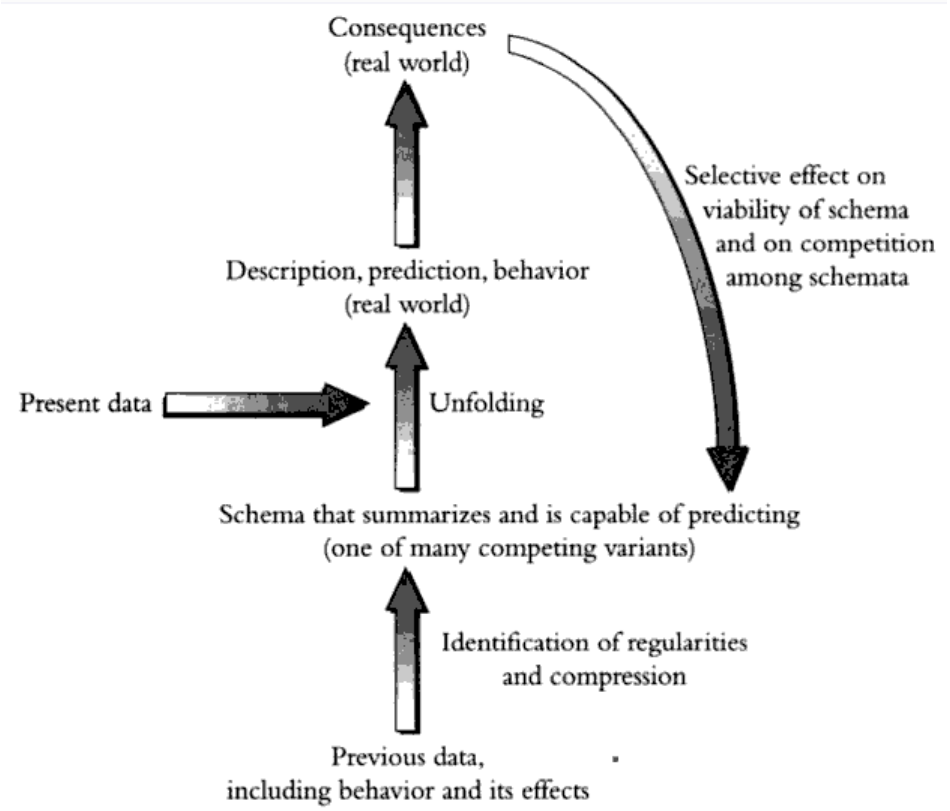


Figure 88: How a Complex System Works [1]

For instance, with the Iraq context, extreme-case data served as the "Previous data" from which HTM networks identify "regularities and compression" with hierarchical learning. The output from the compression is a schema, which in the case of HTM is the set of top-level node's Markov-chains. We then examined "Present data" on a monthly basis about the actual Iraq context with this schema. We did not explore prediction in detail (Level 3 SA), but we used the network's feed-forward inference (λ_t) on this data to exert selective pressures on a certain schema over others. For instance, we saw that the schema for recognizing stability consistently suffered problems, whereas the one for instability did not. But the one for recognizing gradations of instability seemed to hold across many implementations. In a similar way, we formed several schemas about the shockwaves context.

Maintaining the Big Picture and the Decision-Making Impact

It is important to note that the information flow does not isolate any element of the complex system. Recall Bar Yam's earlier advice [2]: "Don't take it apart [and] Don't assume only a few parameters are important." When we have trained a network, we have done so with all metrics that are considered important for the system's analysis. Although we are limited in what parameters are available, we consider them all to be equally important. This has been reflected by their equal treatment in terms of normalization.

But there may be some conflict with this approach, in light of shifting decision-making goals. Specifically, our consideration of all variables may drive us towards a solution that is not perfectly aligned with modified goals. For instance, for a complex system like Iraq, we have to specify what about it is important. This narrows the focus to pertinent data (Level 1 SA). Then machine-learning algorithms can work on this data (Level 2 SA). Consequently, this choice of data constrains how the network perceives stability. For example, we include coalition troop strength as an observable and make a monotonic case such that higher troop levels indicate greater stability. But, what if we want lower troop levels *and* stability? In fact, as of the time of writing, the primary goal of the DoD, as a representative of the U.S. Government, was to attain stability with as little U.S. presence (e.g., troops) as possible [130]. In such a case, it would be necessary to train and infer off of data that does *not* include this metric. This is only one example of many possible others about how data constrains recognition. Consequently, this is a crucial issue to consider, especially if goals change, as they tend to do with decision-makers.

We see from this demonstration a potential strong interplay between decision-making and complex system analysis. In doing so, we may have pinpointed an important general rule for the analysis of a complex system: it is important to acknowledge the goal of the analysis because this determines *how* and *what* information is treated. This has been our major result for complex systems research. Of course, this is not surprising

because SA research told us that an operator's goals focused attention onto pertinent data about a complex system. In our work, the decision about what data is pertinent has been somewhat assumed for demonstration purposes. Nevertheless, the importance of the choice of data cannot be underestimated, especially with regards to decision-making affecting the system.

The Role of Biological Inspiration

We have explicitly embraced the perspective here that biological systems can help to understand complex systems. Bar Yam and Gell-Mann pointed us in this direction and some of their examples (e.g., attractor networks and visual system analogies) led us to HTM networks. Recall from earlier that coincidence patterns (C) and Markov-chains (G) in each node re-create certain aspects to cortical circuitry. We have showed how this approach to information condensation in a hierarchy can help the analysis of two complex systems. Each node condensed information about its local receptive field and through feed-forward inference it propagated categorization data up to higher levels of the network. George shows how this is similar to the way cortical circuitry learns to recognize patterns in a constant data stream. We have then taken this idea and extended it to the data stream about a complex system. Gell-Mann and others have repeatedly highlighted the importance of the information, so this was a natural connection to make.

Summary of Addressed Issues

We have briefly summarized here the ways that various ideas from complex systems research have contributed to the HTM-based SA approach. Each of these issues became more developed as we explored ways of implementing them. For instance, our interest in schema formation led us to Endsley and others in human factors. Similarly, our interest in computational approaches to schema formation led us to machine learning, data mining and data fusion. But the root of these investigations has always been from

complex systems research. Consequently, it seems that the work done in the previous chapter is a significant result for complex systems research. As Gell-Mann told us, complex systems research is a fight-fire-with-fire discipline. In our approach, we have analyzed the complex systems of rudimentary high-speed flow and the Iraq War with a computerized complex system (an HTM network).

Contributions to Situational Awareness Research?

There have not been contributions to SA research as there have been for other disciplines. Instead, we have built on the foundation provided by this research to see how it translates to the processing mechanisms of modern computers. For instance, Endsley's model of SA divided it into three levels and this guidance has been instrumental. This framework provided a guide for the information flow we have used in our implementation. Furthermore, it points the way forward so that future implementations can possibly move into the abilities of Level 3 SA, which is where prediction happens. With this thought in mind, the next and final chapter addresses some choices that could have been made differently and how this research may lead the way to future research on computational SA.

CHAPTER 8

RECOMMENDATIONS

Along the course of our research, there have been paths not taken and choices made to stay on topic. For instance, the computational implementation of Level 3 SA has not been pursued. Also, we have chosen to focus on HTM for our machine learning needs, rather than fully examining other options. By doing so, there are future paths of research that we have opened. So first, we shall account for some choices made and the lessons we have learned from them. Secondly, we will discuss what could have been done differently, in light of these lessons. Finally, we will speculate on some future research paths stemming from this work.

Choices Made and Lessons Learned

It is not possible to recount each of the choices made in the course of this research, but we can certainly summarize the key points. First, we have approached complex system analysis from the perspective of biological systems. Specifically, we have assumed that there is something non-trivial about the way biological systems interpret and interact with their world. Having chosen this path, we have focused on the processing mechanisms of SA and then chosen to implement them with HTM networks.

But this was not the only path available. For instance, it is possible to use other techniques involving neural networks (e.g., [38]-[45]) or hierarchical hidden Markov models [69] to generate some kind of SA of a context's data. These techniques do not claim as direct influence from biological systems as HTM does.

Furthermore, just because HTM has such heritage does not imply that it is a direct replication of a human brain. So if there has been any lesson learned here then it is that great care must be taken not to over-extend the analogy. Just because an HTM network's

processing structure resembles that of human cortical anatomy does not imply that the feed-forward inference is meaningful. Rather, it is necessary to use careful training and validation strategies – such as, extreme-case bounding – to test the meaning of the feed-forward inference.

Another choice that has been made in this research is to pursue comprehension of a partially occluded system. For the shockwaves context, the fact that we were ignoring viscosity and Prandtl number were nearly irrelevant [114][116]-[118]. These properties of the flow could justifiably be occluded for equilibrium flow analysis. But for the Iraq context, we only tracked *N* of a possibly huge number of metrics that can be used to describe the Iraq War during 2003-2008. We got some sense of how SA changed as the number of metrics changed, but all the while we were dealing with a partially occluded observation. Based on the results, it seems that one lesson learned has been that it is possible to form computational SA from an ill-posed problem. But the tricky part is determining its significance. For instance, when a network was trained on the actual Iraq context data, we found it difficult to gauge the significance of its top-level node's *C* and *G*. It was only when we trained on the extreme-case data that we were able to assess the significance of top-level *C* and *G*. Then we were able to do the same with actual data on the Iraq context.

What Could Be Done Differently

Once again, there are many things that could have been done differently, so we will focus on key points in the approach rather than small details. The lessons we have pulled from the research provide a guide in doing so. One thing to do differently would be to compare other computational SA approaches in more detail. How would a CARTMAP neural network like the one used in one baseline work on the Iraq context data? How would its fusion compare to that of an HTM network? Comparisons like this

could have been done more in-depth. They were not ignored completely though, since we considered rudimentary categorization algorithms based on Euclidean distance.

Another thing to do differently is to probe this occlusion issue. Occlusion is an active area of research in image and video analysis [121]-[123] and drawing on expertise from this, and other disciplines, might further the understanding on this point. Furthermore, building on the occlusion issue, it might have been possible to see how SA changes as more or fewer metrics are added in more of an automatic fashion. For instance, we have manually reduced the number of metrics and assessed SA formation. But a key aspect to SA maintenance is the ability to flexibly move between pertinent metrics' perception as goals change. We saw earlier how the goals affect perception and consequently this overall interaction could have been more sophisticated. Nevertheless, substantial results have been found with the approach done here. So while these things could have been alternative ways of doing the research done and reported here, the fact that we have not done them does not invalidate what has been done here.

Future Paths of Research

The exciting thing about this research is that there are many paths to follow. Sometimes the breadth of possibilities is daunting though and it becomes necessary to bound the problem somehow so that progress can be made. We have had to do just that to get some firm results on this computational approach to SA. But there are other examples that can be tested. For instance, this approach to computational SA only requires a temporally structured data set and the set of goals one has when analyzing this data. With that general boundary, there are possible applications for computational SA in quantitative finance, weather analysis and economics, to name a few. In fact, wherever there is a complex system there is a potential example on which this approach to computational SA can be formed and tested. As with our contexts, the specific ways it is

formed and tested will vary slightly between different contexts. But the general question is the same (see Figure 16): how do we get from the data to the understanding?

The other significant class of paths that has yet to be explored in real detail is the control-loop aspect. We only thinly described decision-loops with respect to this computational SA. That is because our focus has been on the approach to computational SA, rather than its use. The use – decision-making – is what motivated the importance of the approach. So a future research path would be to see how the computational SA could be put inside an automated decision-loop. Consequently, there could be interaction between the SA and the decisions, as there is for real-life biological systems. In exploring this, we would not only gain insight into new ways of creating control systems. But we might also gain insight into the biological mechanisms that have evolved to provide such wonderful abilities to natural creatures.

APPENDIX A

RIVER (WAVES) CONTEXT DATA

This appendix contains data pertinent to the river (waves) context experiment. The exact network parameters used for this network are shown below. This is an unsupervised version of the original waves network included with NuPIC, except there are slight modifications made to the hierarchy. These changes can be seen by comparing the original supervised network to the one whose parameters are shown below.

A.1. Unsupervised River (Waves) Network Parameter Settings

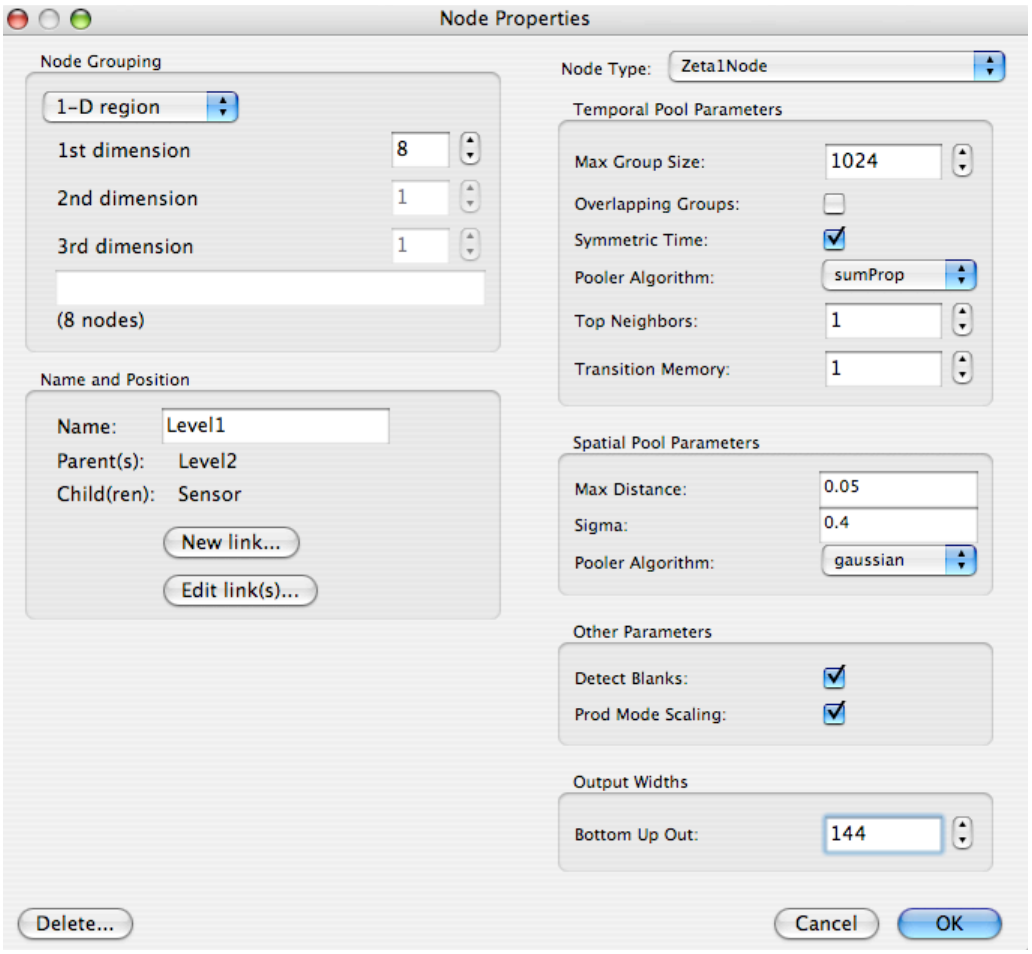


Figure 89: Bottom-Level Node Settings

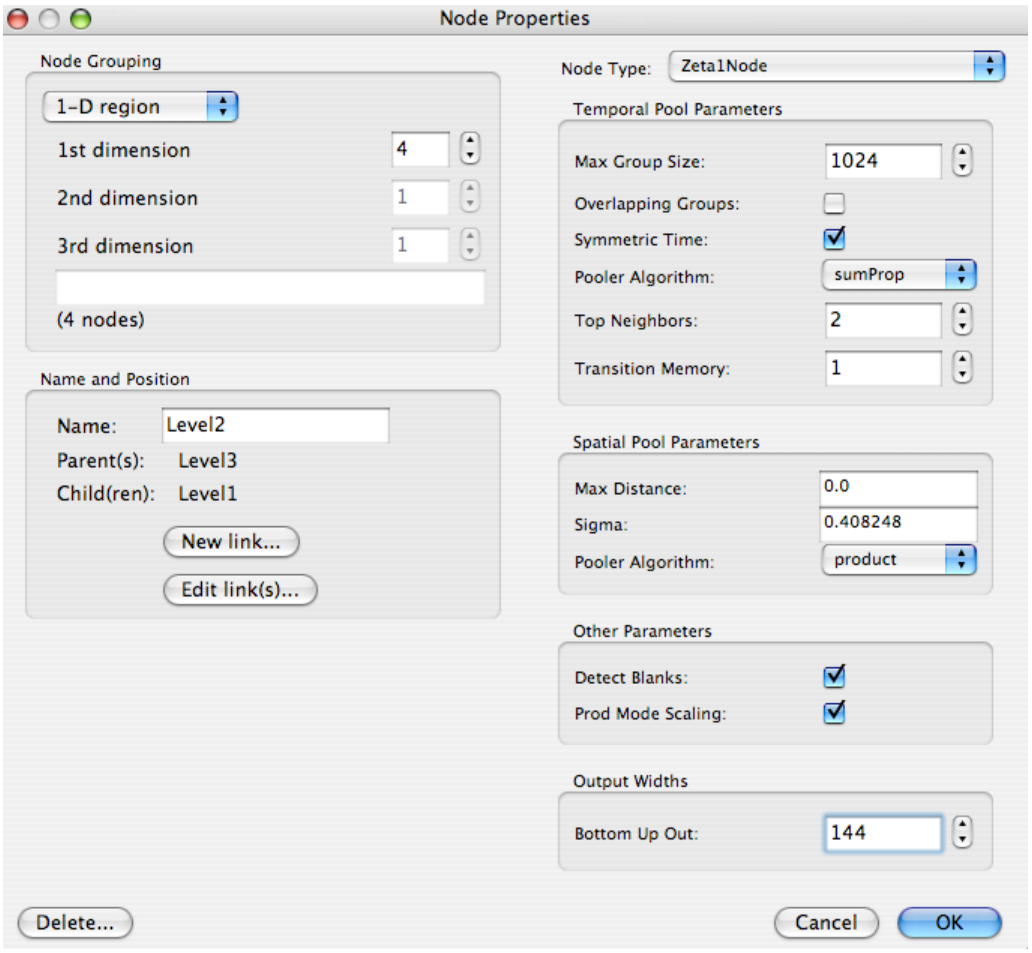


Figure 90: Second-Level Node Settings

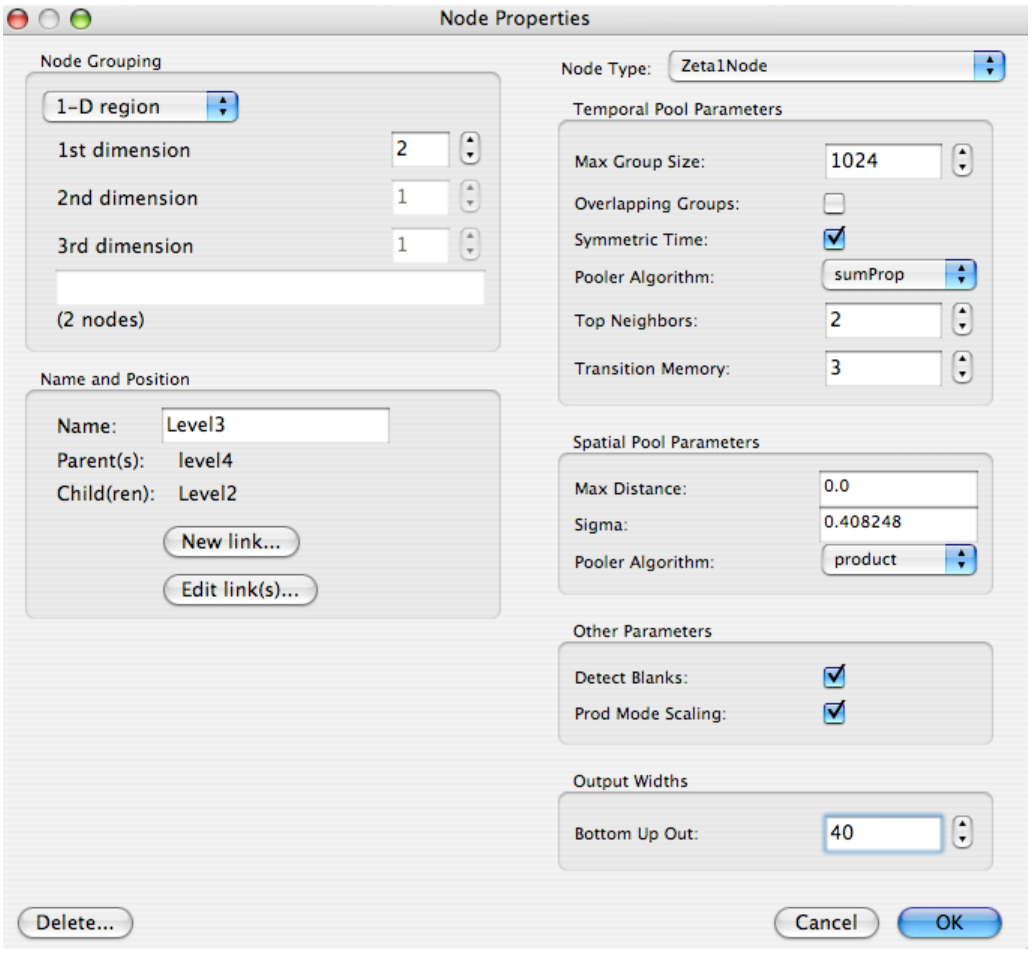


Figure 91: Third-Level Node Settings

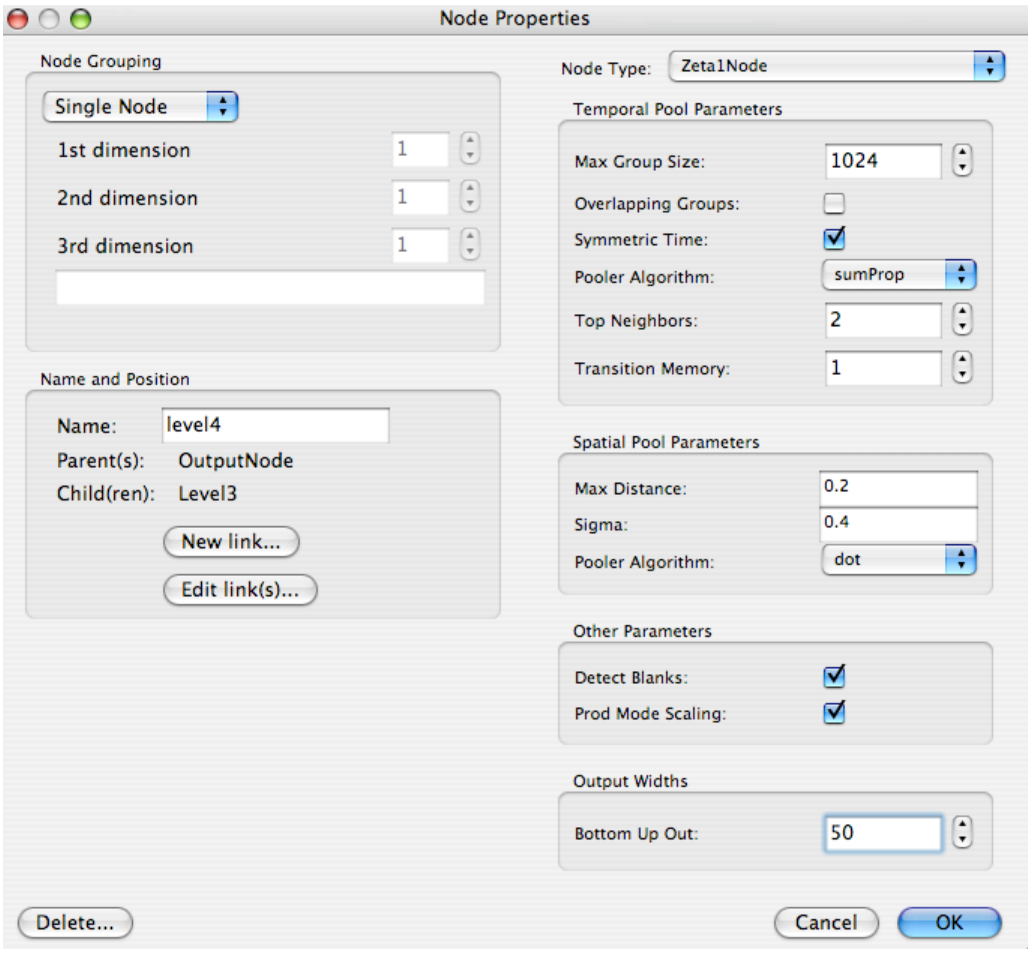


Figure 92: Top-Level Node Settings

APPENDIX B

SHOCKWAVES CONTEXT DATA

This appendix contains data pertinent to the shockwaves context experiments. Each section title indicates the contents. In this appendix, the following will be found: training and testing data, network parameters and network inference results. The following data has been used either to train or test HTM networks used in developing the SA of the shockwaves context. Similarly, the network parameters that follow have also been used to create SA of the shockwaves context. The inference results below document a large amount of this recognition capability.

The title of each section indicates the characteristic information to be found in it. These sections are labeled and numbered to correspond to the main body of the text. For example, the "*N = 2* log-transformed training dataset" refers to the data of temperature (*T*) and flow speed (*u*) transformed by a base-10 logarithm used for training. Similarly, when a network's parameters are given, the title of the section indicates how its information relates to the text. For instance, "Unsupervised Network Settings for *N = 2*" refers to the parameters of the network used to analyze temperature (*T*) and flow speed (*u*). Inference data is similarly labeled in accordance with its appearance in the main body of the text.

B.1. *N = 4* Training Dataset

time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
0	1.28E+05	1.48	2.74E+05	411.91
1	2.24E+05	2.40	3.27E+05	253.19
2	2.24E+05	2.40	3.27E+05	253.19
3	2.24E+05	2.40	3.27E+05	253.19
4	2.24E+05	2.40	3.27E+05	253.19
5	2.24E+05	2.40	3.27E+05	253.19
6	2.24E+05	2.40	3.27E+05	253.19
7	2.24E+05	2.40	3.27E+05	253.19
8	2.24E+05	2.40	3.27E+05	253.19
9	2.24E+05	2.40	3.27E+05	253.19
10	2.24E+05	2.40	3.27E+05	377.88
11	2.51E+05	2.61	3.37E+05	348.48
12	2.51E+05	2.61	3.37E+05	348.48
13	2.51E+05	2.61	3.37E+05	348.48
14	2.51E+05	2.61	3.37E+05	348.48
15	2.51E+05	2.61	3.37E+05	348.48
16	2.51E+05	2.61	3.37E+05	348.48
17	2.51E+05	2.61	3.37E+05	348.48
18	2.51E+05	2.61	3.37E+05	348.48
19	2.51E+05	2.61	3.37E+05	348.48
20	2.51E+05	2.61	3.37E+05	519.51
21	5.45E+05	4.47	4.26E+05	303.10
22	5.45E+05	4.47	4.26E+05	303.10
23	5.45E+05	4.47	4.26E+05	303.10
24	5.45E+05	4.47	4.26E+05	303.10
25	5.45E+05	4.47	4.26E+05	303.10
26	5.45E+05	4.47	4.26E+05	303.10
27	5.45E+05	4.47	4.26E+05	303.10
28	5.45E+05	4.47	4.26E+05	303.10
29	5.45E+05	4.47	4.26E+05	303.10
30	5.45E+05	4.47	4.26E+05	769.29
31	2.11E+06	10.98	6.73E+05	313.00
32	2.11E+06	10.98	6.73E+05	313.00
33	2.11E+06	10.98	6.73E+05	313.00
34	2.11E+06	10.98	6.73E+05	313.00
35	2.11E+06	10.98	6.73E+05	313.00
36	2.11E+06	10.98	6.73E+05	313.00
37	2.11E+06	10.98	6.73E+05	313.00
38	2.11E+06	10.98	6.73E+05	313.00
39	2.11E+06	10.98	6.73E+05	313.00
40	2.11E+06	10.98	6.73E+05	929.24
41	7.55E+06	25.75	1.03E+06	396.42
42	7.55E+06	25.75	1.03E+06	396.42
43	7.55E+06	25.75	1.03E+06	396.42
44	7.55E+06	25.75	1.03E+06	396.42
45	7.55E+06	25.75	1.03E+06	396.42
46	7.55E+06	25.75	1.03E+06	396.42
47	7.55E+06	25.75	1.03E+06	396.42
48	7.55E+06	25.75	1.03E+06	396.42
49	7.55E+06	25.75	1.03E+06	396.42
50	7.55E+06	25.75	1.03E+06	880.15
51	1.54E+07	42.32	1.27E+06	535.47
52	1.54E+07	42.32	1.27E+06	535.47
53	1.54E+07	42.32	1.27E+06	535.47
54	1.54E+07	42.32	1.27E+06	535.47
55	1.54E+07	42.32	1.27E+06	535.47
56	1.54E+07	42.32	1.27E+06	535.47
57	1.54E+07	42.32	1.27E+06	535.47
58	1.54E+07	42.32	1.27E+06	535.47
59	1.54E+07	42.32	1.27E+06	535.47

time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
60	1.54E+07	42.32	1.27E+06	1398.65
61	6.64E+07	110.45	2.11E+06	535.90
62	6.64E+07	110.45	2.11E+06	535.90
63	6.64E+07	110.45	2.11E+06	535.90
64	6.64E+07	110.45	2.11E+06	535.90
65	6.64E+07	110.45	2.11E+06	535.90
66	6.64E+07	110.45	2.11E+06	535.90
67	6.64E+07	110.45	2.11E+06	535.90
68	6.64E+07	110.45	2.11E+06	535.90
69	6.64E+07	110.45	2.11E+06	535.90
70	6.64E+07	110.45	2.11E+06	1717.26
71	2.60E+08	272.99	3.34E+06	694.81
72	2.60E+08	272.99	3.34E+06	694.81
73	2.60E+08	272.99	3.34E+06	694.81
74	2.60E+08	272.99	3.34E+06	694.81
75	2.60E+08	272.99	3.34E+06	694.81
76	2.60E+08	272.99	3.34E+06	694.81
77	2.60E+08	272.99	3.34E+06	694.81
78	2.60E+08	272.99	3.34E+06	694.81
79	2.60E+08	272.99	3.34E+06	694.81
80	2.60E+08	272.99	3.34E+06	1220.52
81	2.98E+08	300.36	3.47E+06	1109.28
82	2.98E+08	300.36	3.47E+06	1109.28
83	2.98E+08	300.36	3.47E+06	1109.28
84	2.98E+08	300.36	3.47E+06	1109.28
85	2.98E+08	300.36	3.47E+06	1109.28
86	2.98E+08	300.36	3.47E+06	1109.28
87	2.98E+08	300.36	3.47E+06	1109.28
88	2.98E+08	300.36	3.47E+06	1109.28
89	2.98E+08	300.36	3.47E+06	1109.28
90	2.98E+08	300.36	3.47E+06	2296.34
91	1.27E+09	778.29	5.71E+06	886.21
92	1.27E+09	778.29	5.71E+06	886.21
93	1.27E+09	778.29	5.71E+06	886.21
94	1.27E+09	778.29	5.71E+06	886.21
95	1.27E+09	778.29	5.71E+06	886.21
96	1.27E+09	778.29	5.71E+06	886.21
97	1.27E+09	778.29	5.71E+06	886.21
98	1.27E+09	778.29	5.71E+06	886.21
99	1.27E+09	778.29	5.71E+06	886.21

B.2. *N = 4* Testing Dataset: 5-10% Perturbation of Training Dataset

time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
0	1.20E+05	1.58	2.92E+05	448.17
1	2.04E+05	2.17	3.53E+05	238.87
2	2.05E+05	2.61	3.44E+05	239.92
3	2.45E+05	2.60	3.53E+05	238.92
4	2.39E+05	2.28	3.09E+05	232.03
5	2.42E+05	2.16	3.10E+05	238.95
6	2.05E+05	2.28	2.96E+05	238.43
7	2.08E+05	2.25	3.59E+05	233.07
8	2.37E+05	2.24	3.47E+05	235.07
9	2.43E+05	2.27	3.51E+05	235.71
10	2.45E+05	2.26	3.45E+05	357.74
11	2.35E+05	2.36	3.14E+05	370.63
12	2.29E+05	2.75	3.66E+05	319.90
13	2.71E+05	2.46	3.65E+05	375.51
14	2.67E+05	2.45	3.21E+05	325.63
15	2.31E+05	2.43	3.17E+05	370.81
16	2.38E+05	2.80	3.56E+05	367.36
17	2.33E+05	2.77	3.65E+05	381.38
18	2.73E+05	2.41	3.16E+05	372.14
19	2.37E+05	2.42	3.17E+05	318.20
20	2.75E+05	2.81	3.55E+05	561.00
21	5.77E+05	4.19	4.53E+05	320.05
22	5.00E+05	4.05	4.02E+05	318.84
23	5.92E+05	4.70	3.96E+05	326.82
24	5.02E+05	4.78	4.69E+05	322.56
25	4.95E+05	4.90	4.49E+05	321.47
26	5.79E+05	4.71	4.51E+05	282.69
27	4.90E+05	4.90	3.97E+05	278.86
28	4.93E+05	4.84	4.62E+05	320.24
29	5.84E+05	4.72	4.49E+05	324.17
30	5.15E+05	4.18	4.66E+05	810.87
31	2.32E+06	11.84	7.15E+05	334.14
32	1.91E+06	10.24	7.24E+05	331.42
33	2.28E+06	12.06	6.07E+05	295.91
34	2.30E+06	10.18	6.12E+05	285.84
35	1.97E+06	10.39	7.17E+05	295.92
36	2.26E+06	11.65	6.32E+05	284.65
37	2.24E+06	11.82	6.27E+05	284.31
38	2.30E+06	11.88	7.39E+05	340.94
39	1.95E+06	11.70	6.22E+05	331.19
40	2.30E+06	12.00	6.35E+05	987.68
41	6.95E+06	23.42	9.44E+05	366.86
42	8.17E+06	24.24	1.08E+06	376.35
43	8.10E+06	27.08	1.11E+06	362.39
44	6.83E+06	27.55	1.10E+06	371.04
45	8.28E+06	24.10	1.13E+06	368.25
46	8.13E+06	28.12	1.10E+06	418.21
47	6.87E+06	27.50	1.09E+06	362.52
48	7.17E+06	24.22	9.61E+05	360.80
49	8.05E+06	23.70	1.10E+06	421.33
50	7.04E+06	27.86	9.63E+05	792.50
51	1.43E+07	40.09	1.19E+06	573.94
52	1.44E+07	45.80	1.36E+06	582.37
53	1.41E+07	39.63	1.17E+06	581.11
54	1.64E+07	45.29	1.34E+06	483.09
55	1.44E+07	45.97	1.38E+06	564.17
56	1.40E+07	44.99	1.17E+06	562.32
57	1.68E+07	46.19	1.38E+06	573.29
58	1.44E+07	45.46	1.16E+06	481.99
59	1.67E+07	39.26	1.34E+06	501.48

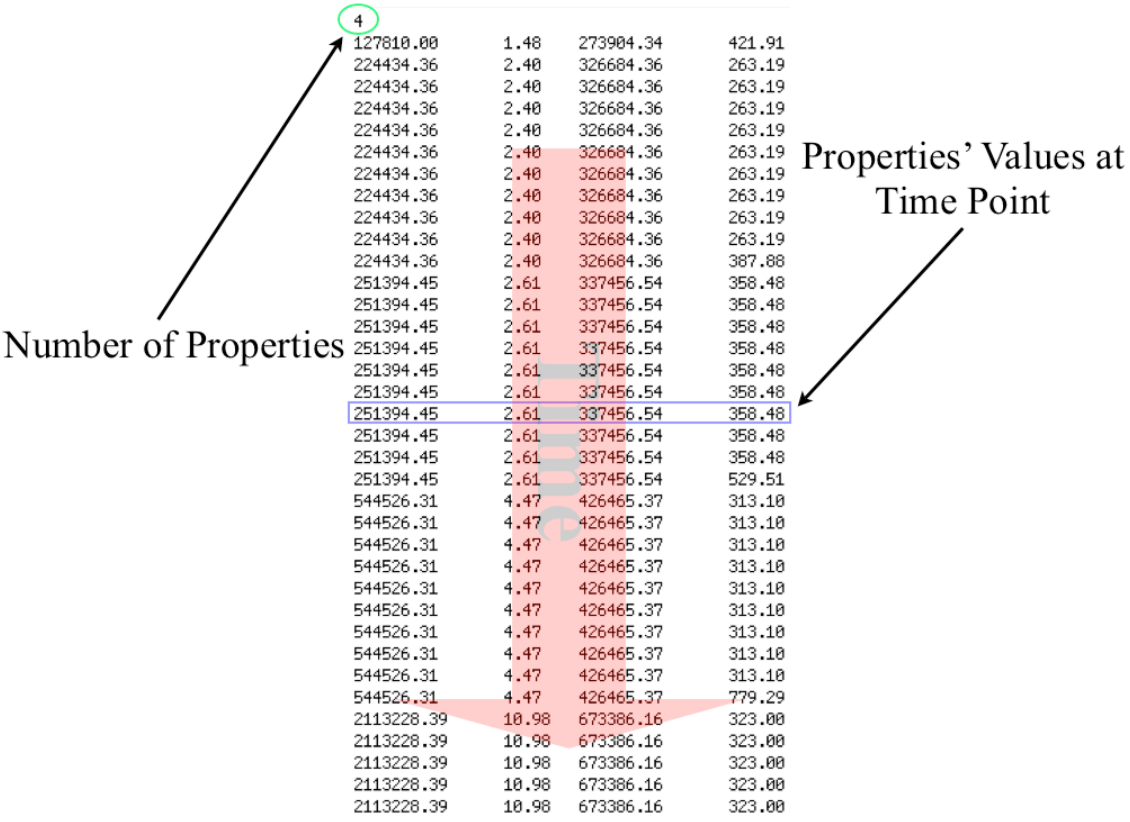
time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
60	1.44E+07	45.66	1.39E+06	1279.00
61	6.28E+07	100.16	1.93E+06	572.00
62	7.01E+07	101.96	2.28E+06	491.23
63	7.09E+07	118.43	1.99E+06	587.89
64	6.15E+07	118.05	1.99E+06	499.10
65	6.11E+07	117.32	1.95E+06	567.70
66	6.21E+07	102.88	2.26E+06	577.95
67	6.10E+07	103.66	1.98E+06	565.33
68	7.10E+07	119.24	1.99E+06	579.29
69	6.23E+07	117.52	1.98E+06	565.20
70	7.29E+07	101.24	2.00E+06	1556.90
71	2.79E+08	291.62	3.15E+06	655.17
72	2.35E+08	292.86	3.56E+06	634.15
73	2.78E+08	299.65	3.02E+06	743.05
74	2.34E+08	246.82	3.60E+06	756.21
75	2.43E+08	287.21	3.56E+06	739.58
76	2.40E+08	251.26	3.11E+06	656.42
77	2.43E+08	293.78	3.16E+06	649.13
78	2.43E+08	245.86	3.64E+06	645.42
79	2.38E+08	246.81	3.06E+06	631.85
80	2.39E+08	255.92	3.05E+06	1158.10
81	2.78E+08	284.07	3.81E+06	1025.73
82	3.26E+08	278.76	3.15E+06	1044.24
83	2.82E+08	323.15	3.76E+06	1037.87
84	3.19E+08	282.42	3.28E+06	1211.33
85	3.14E+08	284.82	3.77E+06	998.45
86	2.75E+08	279.83	3.18E+06	1014.46
87	3.13E+08	277.71	3.74E+06	1015.22
88	2.78E+08	326.03	3.72E+06	1185.59
89	3.22E+08	324.79	3.81E+06	1052.93
90	2.77E+08	276.53	3.13E+06	2421.30
91	1.21E+09	846.96	5.20E+06	831.43
92	1.34E+09	837.95	5.38E+06	950.70
93	1.17E+09	705.88	6.02E+06	948.01
94	1.36E+09	836.22	6.17E+06	942.52
95	1.18E+09	846.01	6.17E+06	963.67
96	1.39E+09	845.07	6.18E+06	821.31
97	1.39E+09	828.63	6.21E+06	811.54
98	1.20E+09	829.69	5.29E+06	803.31
99	1.37E+09	845.36	5.17E+06	799.86

B.3. *N = 4* Testing Dataset: Perturbation by Constant Value of *10* (except to ρ)

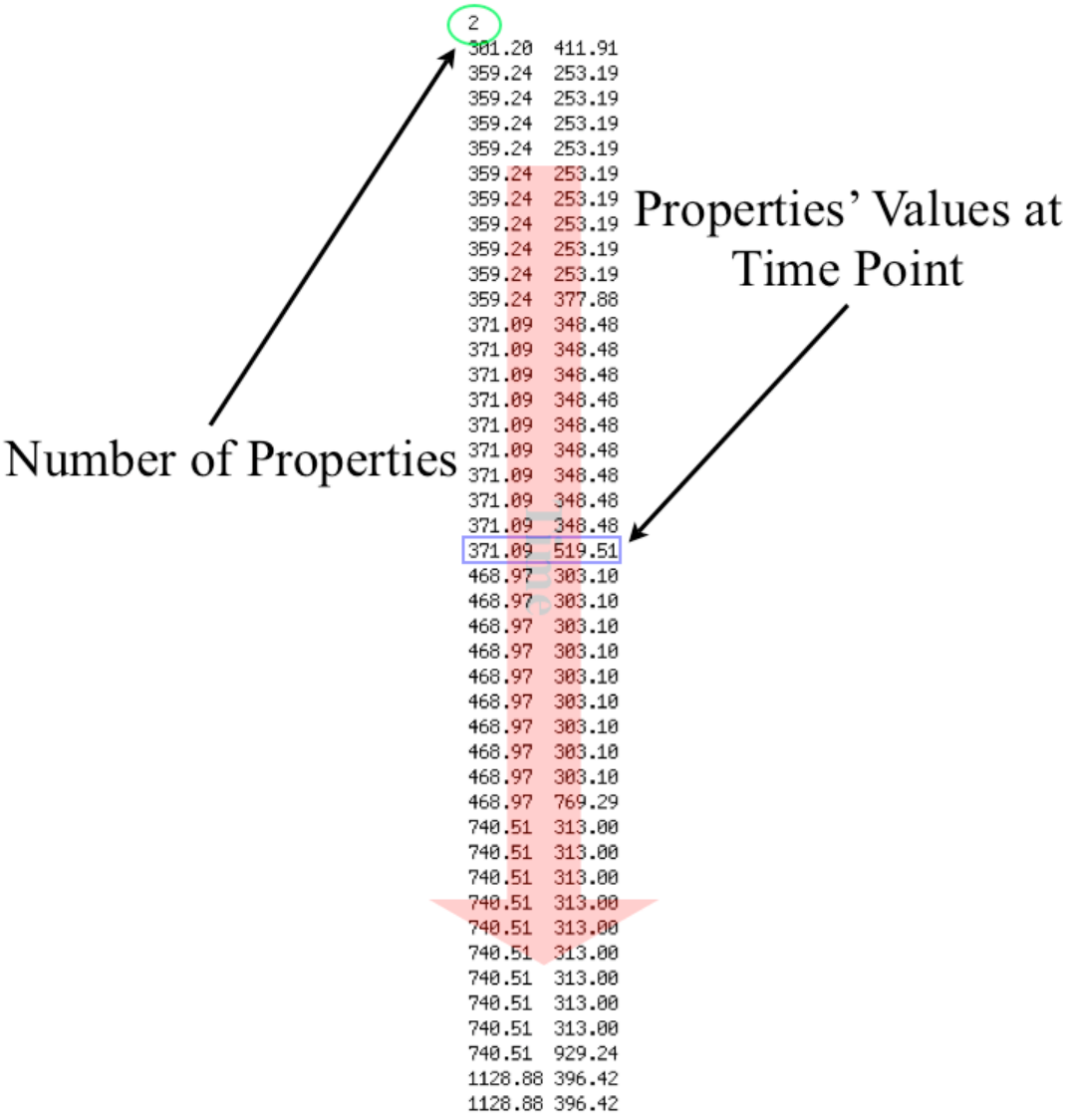
time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
0	1.28E+05	1.48	2.74E+05	421.91
1	2.24E+05	2.40	3.27E+05	263.19
2	2.24E+05	2.40	3.27E+05	263.19
3	2.24E+05	2.40	3.27E+05	263.19
4	2.24E+05	2.40	3.27E+05	263.19
5	2.24E+05	2.40	3.27E+05	263.19
6	2.24E+05	2.40	3.27E+05	263.19
7	2.24E+05	2.40	3.27E+05	263.19
8	2.24E+05	2.40	3.27E+05	263.19
9	2.24E+05	2.40	3.27E+05	263.19
10	2.24E+05	2.40	3.27E+05	387.88
11	2.51E+05	2.61	3.37E+05	358.48
12	2.51E+05	2.61	3.37E+05	358.48
13	2.51E+05	2.61	3.37E+05	358.48
14	2.51E+05	2.61	3.37E+05	358.48
15	2.51E+05	2.61	3.37E+05	358.48
16	2.51E+05	2.61	3.37E+05	358.48
17	2.51E+05	2.61	3.37E+05	358.48
18	2.51E+05	2.61	3.37E+05	358.48
19	2.51E+05	2.61	3.37E+05	358.48
20	2.51E+05	2.61	3.37E+05	529.51
21	5.45E+05	4.47	4.26E+05	313.10
22	5.45E+05	4.47	4.26E+05	313.10
23	5.45E+05	4.47	4.26E+05	313.10
24	5.45E+05	4.47	4.26E+05	313.10
25	5.45E+05	4.47	4.26E+05	313.10
26	5.45E+05	4.47	4.26E+05	313.10
27	5.45E+05	4.47	4.26E+05	313.10
28	5.45E+05	4.47	4.26E+05	313.10
29	5.45E+05	4.47	4.26E+05	313.10
30	5.45E+05	4.47	4.26E+05	779.29
31	2.11E+06	10.98	6.73E+05	323.00
32	2.11E+06	10.98	6.73E+05	323.00
33	2.11E+06	10.98	6.73E+05	323.00
34	2.11E+06	10.98	6.73E+05	323.00
35	2.11E+06	10.98	6.73E+05	323.00
36	2.11E+06	10.98	6.73E+05	323.00
37	2.11E+06	10.98	6.73E+05	323.00
38	2.11E+06	10.98	6.73E+05	323.00
39	2.11E+06	10.98	6.73E+05	323.00
40	2.11E+06	10.98	6.73E+05	939.24
41	7.55E+06	25.75	1.03E+06	406.42
42	7.55E+06	25.75	1.03E+06	406.42
43	7.55E+06	25.75	1.03E+06	406.42
44	7.55E+06	25.75	1.03E+06	406.42
45	7.55E+06	25.75	1.03E+06	406.42
46	7.55E+06	25.75	1.03E+06	406.42
47	7.55E+06	25.75	1.03E+06	406.42
48	7.55E+06	25.75	1.03E+06	406.42
49	7.55E+06	25.75	1.03E+06	406.42
50	7.55E+06	25.75	1.03E+06	890.15
51	1.54E+07	42.32	1.27E+06	545.47
52	1.54E+07	42.32	1.27E+06	545.47
53	1.54E+07	42.32	1.27E+06	545.47
54	1.54E+07	42.32	1.27E+06	545.47
55	1.54E+07	42.32	1.27E+06	545.47
56	1.54E+07	42.32	1.27E+06	545.47
57	1.54E+07	42.32	1.27E+06	545.47
58	1.54E+07	42.32	1.27E+06	545.47
59	1.54E+07	42.32	1.27E+06	545.47

time	press	dens	h	u
time step	N/m^2	kg/m^3	J/kg/K	m/s
60	1.54E+07	42.32	1.27E+06	1408.65
61	6.64E+07	110.45	2.11E+06	545.90
62	6.64E+07	110.45	2.11E+06	545.90
63	6.64E+07	110.45	2.11E+06	545.90
64	6.64E+07	110.45	2.11E+06	545.90
65	6.64E+07	110.45	2.11E+06	545.90
66	6.64E+07	110.45	2.11E+06	545.90
67	6.64E+07	110.45	2.11E+06	545.90
68	6.64E+07	110.45	2.11E+06	545.90
69	6.64E+07	110.45	2.11E+06	545.90
70	6.64E+07	110.45	2.11E+06	1727.26
71	2.60E+08	272.99	3.34E+06	704.81
72	2.60E+08	272.99	3.34E+06	704.81
73	2.60E+08	272.99	3.34E+06	704.81
74	2.60E+08	272.99	3.34E+06	704.81
75	2.60E+08	272.99	3.34E+06	704.81
76	2.60E+08	272.99	3.34E+06	704.81
77	2.60E+08	272.99	3.34E+06	704.81
78	2.60E+08	272.99	3.34E+06	704.81
79	2.60E+08	272.99	3.34E+06	704.81
80	2.60E+08	272.99	3.34E+06	1230.52
81	2.98E+08	300.36	3.47E+06	1119.28
82	2.98E+08	300.36	3.47E+06	1119.28
83	2.98E+08	300.36	3.47E+06	1119.28
84	2.98E+08	300.36	3.47E+06	1119.28
85	2.98E+08	300.36	3.47E+06	1119.28
86	2.98E+08	300.36	3.47E+06	1119.28
87	2.98E+08	300.36	3.47E+06	1119.28
88	2.98E+08	300.36	3.47E+06	1119.28
89	2.98E+08	300.36	3.47E+06	1119.28
90	2.98E+08	300.36	3.47E+06	2306.34
91	1.27E+09	778.29	5.71E+06	896.21
92	1.27E+09	778.29	5.71E+06	896.21
93	1.27E+09	778.29	5.71E+06	896.21
94	1.27E+09	778.29	5.71E+06	896.21
95	1.27E+09	778.29	5.71E+06	896.21
96	1.27E+09	778.29	5.71E+06	896.21
97	1.27E+09	778.29	5.71E+06	896.21
98	1.27E+09	778.29	5.71E+06	896.21
99	1.27E+09	778.29	5.71E+06	896.21

B.4. Sample of $N = 4$ Testing Dataset: Space-Separated File Format



B.5. Sample of *N* = *2* Training Dataset: Space-Separated File Format



B.6. *N = 2* Testing Dataset: 5-10% Perturbation of Training Dataset

time	**temp**	**u**
time step	K	m/s
0	277.69	377.72
1	384.91	236.13
2	377.64	240.40
3	331.53	231.45
4	336.90	239.80
5	379.50	237.59
6	379.49	233.21
7	383.64	232.73
8	333.65	235.61
9	384.44	273.76
10	326.39	413.60
11	349.64	366.16
12	395.17	370.44
13	407.00	321.68
14	349.51	365.92
15	394.26	319.06
16	399.54	373.64
17	398.70	378.71
18	405.82	315.94
19	400.67	318.03
20	390.81	556.46
21	499.46	276.01
22	433.02	287.93
23	443.47	326.56
24	502.98	324.20
25	443.06	273.67
26	438.79	320.00
27	431.87	321.76
28	508.76	278.78
29	499.99	281.99
30	426.69	815.44
31	684.57	285.24
32	692.93	331.29
33	688.79	282.96
34	805.37	337.75
35	698.13	328.69
36	684.66	290.65
37	802.63	341.38
38	782.15	331.84
39	698.40	330.84
40	797.51	991.48
41	1207.96	361.18
42	1232.63	431.58
43	1026.58	364.54
44	1221.18	363.52
45	1065.61	373.84
46	1060.15	372.89
47	1210.32	372.02
48	1026.78	369.59
49	1202.67	371.63
50	1188.04	806.43
51	1257.86	502.61
52	1274.72	568.39
53	1473.52	482.19
54	1266.22	504.67
55	1269.45	497.26
56	1282.92	486.72
57	1479.44	502.68
58	1504.77	491.21
59	1298.73	572.77

time	temp	u
time step	K	m/s
60	1274.91	1522.08
61	2164.55	493.34
62	2488.55	581.81
63	2542.80	503.49
64	2474.00	503.19
65	2492.80	499.24
66	2099.82	496.43
67	2150.62	585.86
68	2158.02	579.70
69	2508.36	498.54
70	2175.12	1817.46
71	3414.11	634.04
72	3346.58	744.47
73	3350.63	742.19
74	3454.42	642.71
75	3316.96	762.05
76	3951.62	627.55
77	3313.89	632.42
78	3447.08	759.97
79	3899.72	636.47
80	3875.41	1132.10
81	4177.61	1169.10
82	3532.44	1169.44
83	3505.90	1212.24
84	3581.69	1171.55
85	3546.31	1025.20
86	4012.21	1183.31
87	3540.30	1215.88
88	4194.98	1216.64
89	4032.10	1041.77
90	4036.94	2173.15
91	6611.84	930.77
92	5687.47	813.05
93	5654.84	820.81
94	6636.82	939.71
95	5852.84	945.27
96	6674.07	833.12
97	5754.25	832.07
98	6814.53	971.46
99	6862.69	965.97

B.7. Unsupervised Network Settings for $N = 2$

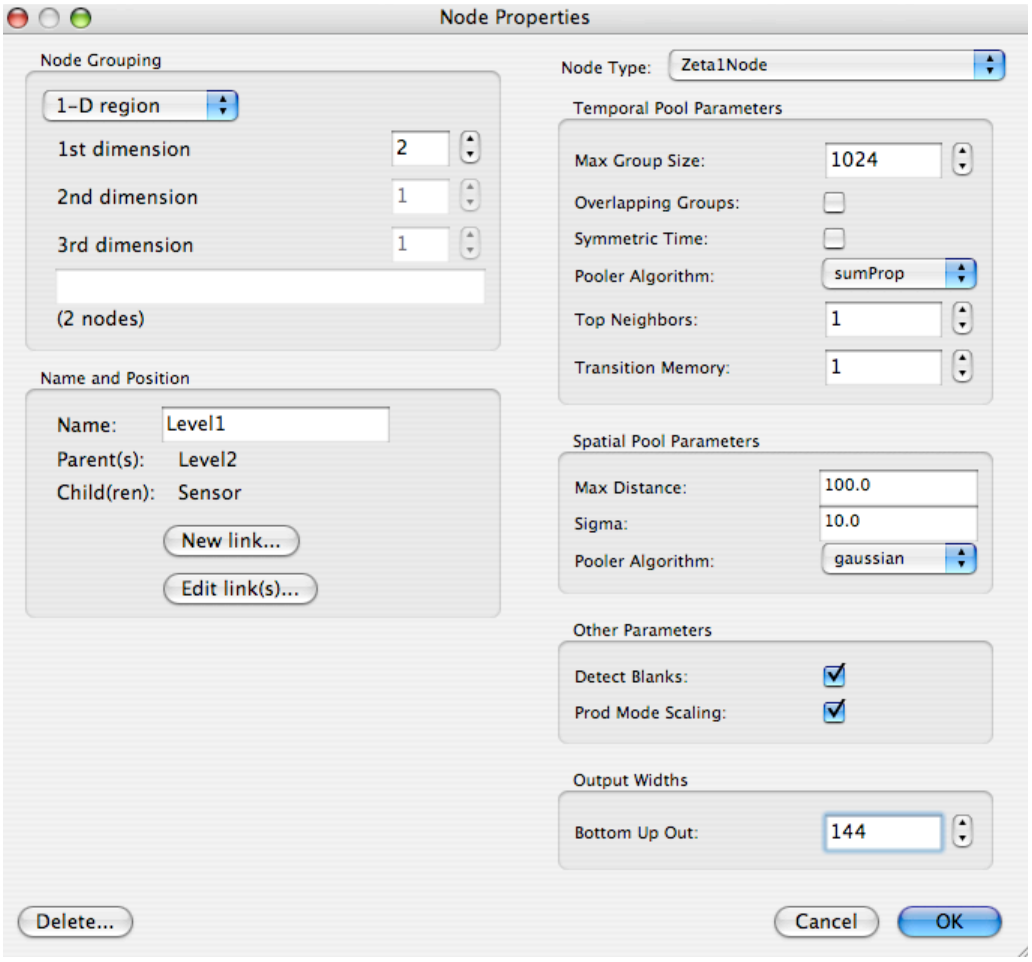


Figure 93: Bottom-Level Node Settings

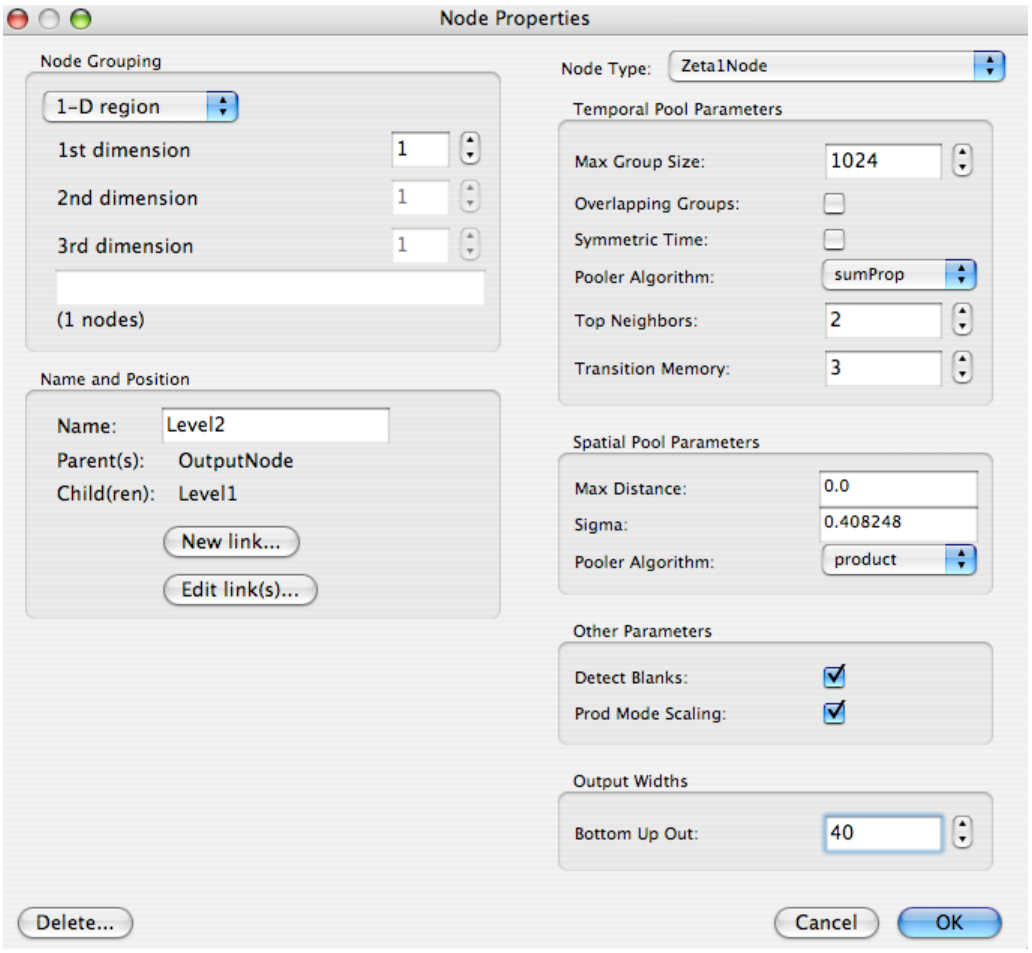


Figure 94: Top-Level Node Settings

B.8. Supervised Network Settings for $N = 2$

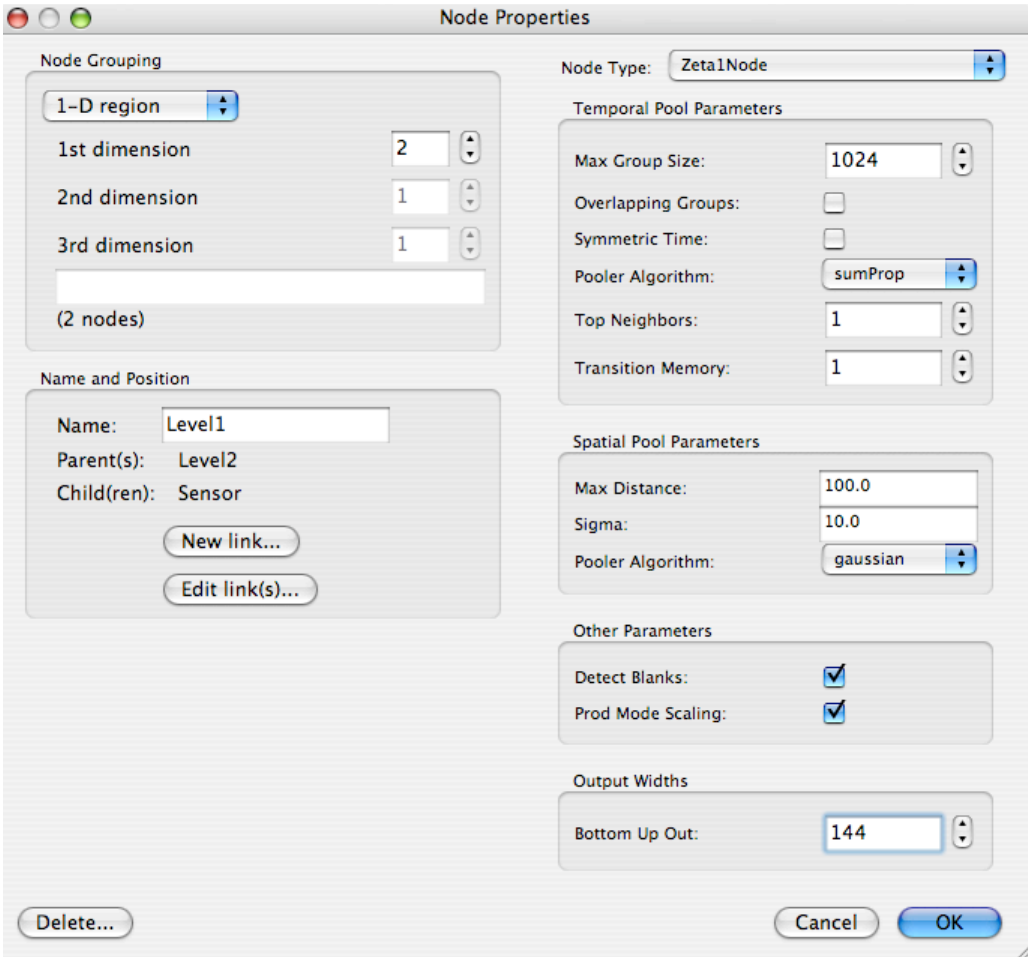


Figure 95: Bottom-Level Node Settings

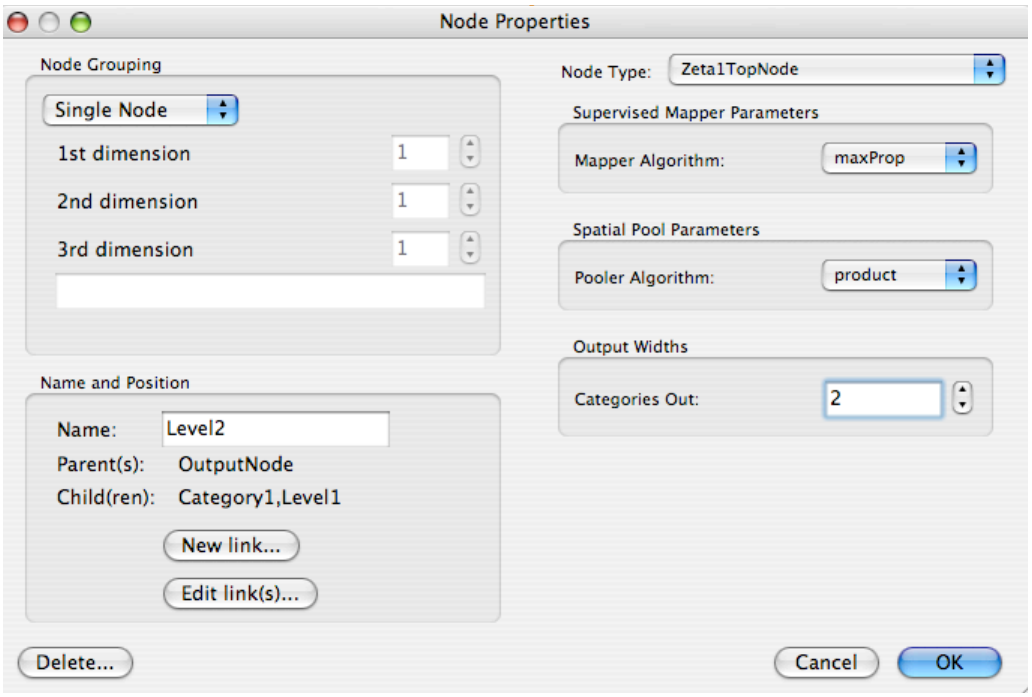


Figure 96: Top-Level Node Settings

B.9. Supervised Network Inference History for $N = 2$ Training Data

Time	First	Second
0	Category 1 1	Category 0 4.535e-20
1	Category 0 1	Category 1 1.733e-34
2	Category 0 1	Category 1 1.733e-34
3	Category 0 1	Category 1 1.733e-34
4	Category 0 1	Category 1 1.733e-34
5	Category 0 1	Category 1 1.733e-34
6	Category 0 1	Category 1 1.733e-34
7	Category 0 1	Category 1 1.733e-34
8	Category 0 1	Category 1 1.733e-34
9	Category 0 1	Category 1 1.733e-34
10	Category 1 1	Category 0 0.006579
11	Category 0 1	Category 1 0.006579
12	Category 0 1	Category 1 0.006579
13	Category 0 1	Category 1 0.006579
14	Category 0 1	Category 1 0.006579
15	Category 0 1	Category 1 0.006579
16	Category 0 1	Category 1 0.006579
17	Category 0 1	Category 1 0.006579
18	Category 0 1	Category 1 0.006579

Time	First	Second
19	Category 0 1	Category 1 0.006579
20	Category 1 1	Category 0 0
21	Category 0 1	Category 1 5.141e-39
22	Category 0 1	Category 1 5.141e-39
23	Category 0 1	Category 1 5.141e-39
24	Category 0 1	Category 1 5.141e-39
25	Category 0 1	Category 1 5.141e-39
26	Category 0 1	Category 1 5.141e-39
27	Category 0 1	Category 1 5.141e-39
28	Category 0 1	Category 1 5.141e-39
29	Category 0 1	Category 1 5.141e-39
30	Category 1 1	Category 0 0
31	Category 0 1	Category 1 0
32	Category 0 1	Category 1 0
33	Category 0 1	Category 1 0
34	Category 0 1	Category 1 0
35	Category 0 1	Category 1 0
36	Category 0 1	Category 1 0
37	Category 0 1	Category 1 0

Time	First	Second
38	Category 0 1	Category 1 0
39	Category 0 1	Category 1 0
40	Category 1 1	Category 0 0
41	Category 0 1	Category 1 0
42	Category 0 1	Category 1 0
43	Category 0 1	Category 1 0
44	Category 0 1	Category 1 0
45	Category 0 1	Category 1 0
46	Category 0 1	Category 1 0
47	Category 0 1	Category 1 0
48	Category 0 1	Category 1 0
49	Category 0 1	Category 1 0
50	Category 1 1	Category 0 0
51	Category 0 1	Category 1 0
52	Category 0 1	Category 1 0
53	Category 0 1	Category 1 0
54	Category 0 1	Category 1 0
55	Category 0 1	Category 1 0
56	Category 0 1	Category 1 0

Time	First	Second
57	Category 0 1	Category 1 0
58	Category 0 1	Category 1 0
59	Category 0 1	Category 1 0
60	Category 1 1	Category 0 0
61	Category 0 1	Category 1 0
62	Category 0 1	Category 1 0
63	Category 0 1	Category 1 0
64	Category 0 1	Category 1 0
65	Category 0 1	Category 1 0
66	Category 0 1	Category 1 0
67	Category 0 1	Category 1 0
68	Category 0 1	Category 1 0
69	Category 0 1	Category 1 0
70	Category 1 1	Category 0 0
71	Category 0 1	Category 1 0
72	Category 0 1	Category 1 0
73	Category 0 1	Category 1 0
74	Category 0 1	Category 1 0
75	Category 0 1	Category 1 0

Time	First	Second
76	Category 0 1	Category 1 0
77	Category 0 1	Category 1 0
78	Category 0 1	Category 1 0
79	Category 0 1	Category 1 0
80	Category 1 1	Category 0 0
81	Category 0 1	Category 1 0
82	Category 0 1	Category 1 0
83	Category 0 1	Category 1 0
84	Category 0 1	Category 1 0
85	Category 0 1	Category 1 0
86	Category 0 1	Category 1 0
87	Category 0 1	Category 1 0
88	Category 0 1	Category 1 0
89	Category 0 1	Category 1 0
90	Category 1 1	Category 0 0
91	Category 0 1	Category 1 0
92	Category 0 1	Category 1 0
93	Category 0 1	Category 1 0
94	Category 0 1	Category 1 0
95	Category 0 1	Category 1 0
96	Category 0 1	Category 1 0
97	Category 0 1	Category 1 0
98	Category 0 1	Category 1 0
99	Category 0 1	Category 1 0

B.10. Unsupervised Network Settings for Log-transformed *N* = *2*

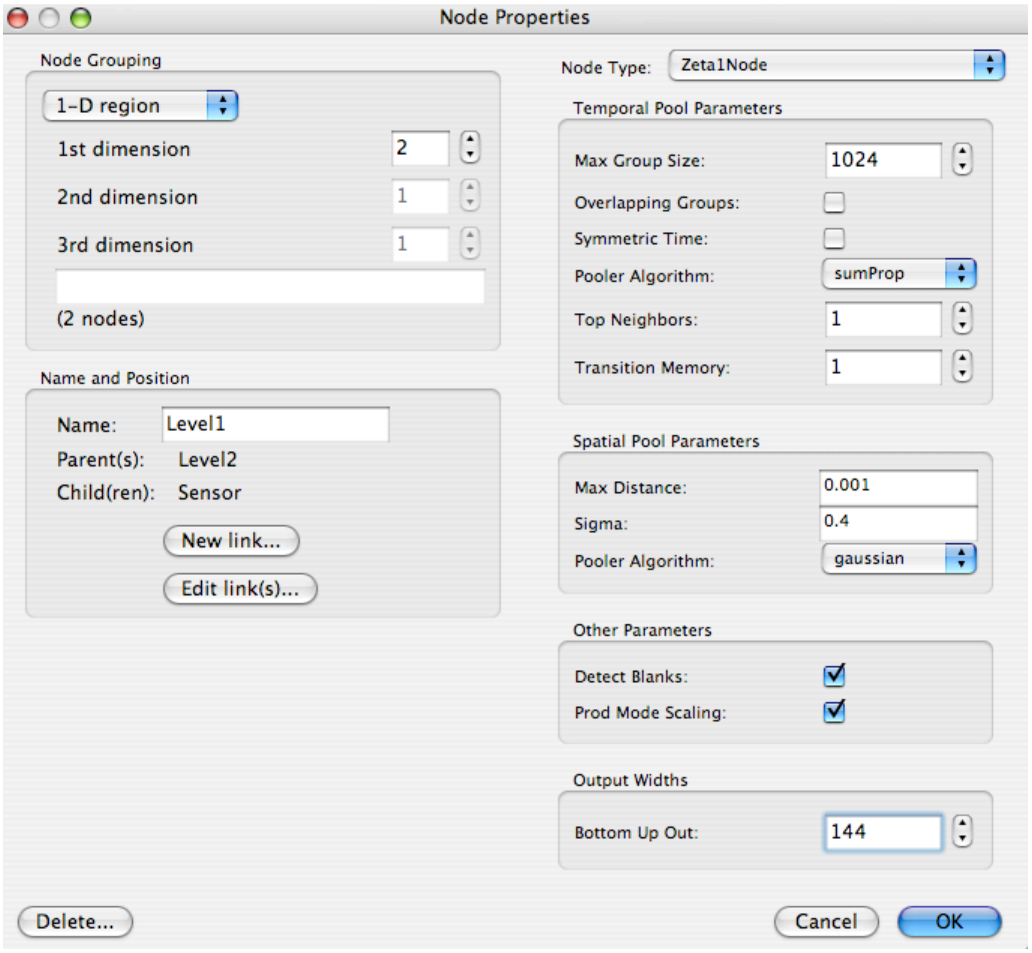


Figure 97: Bottom-Level Node Settings

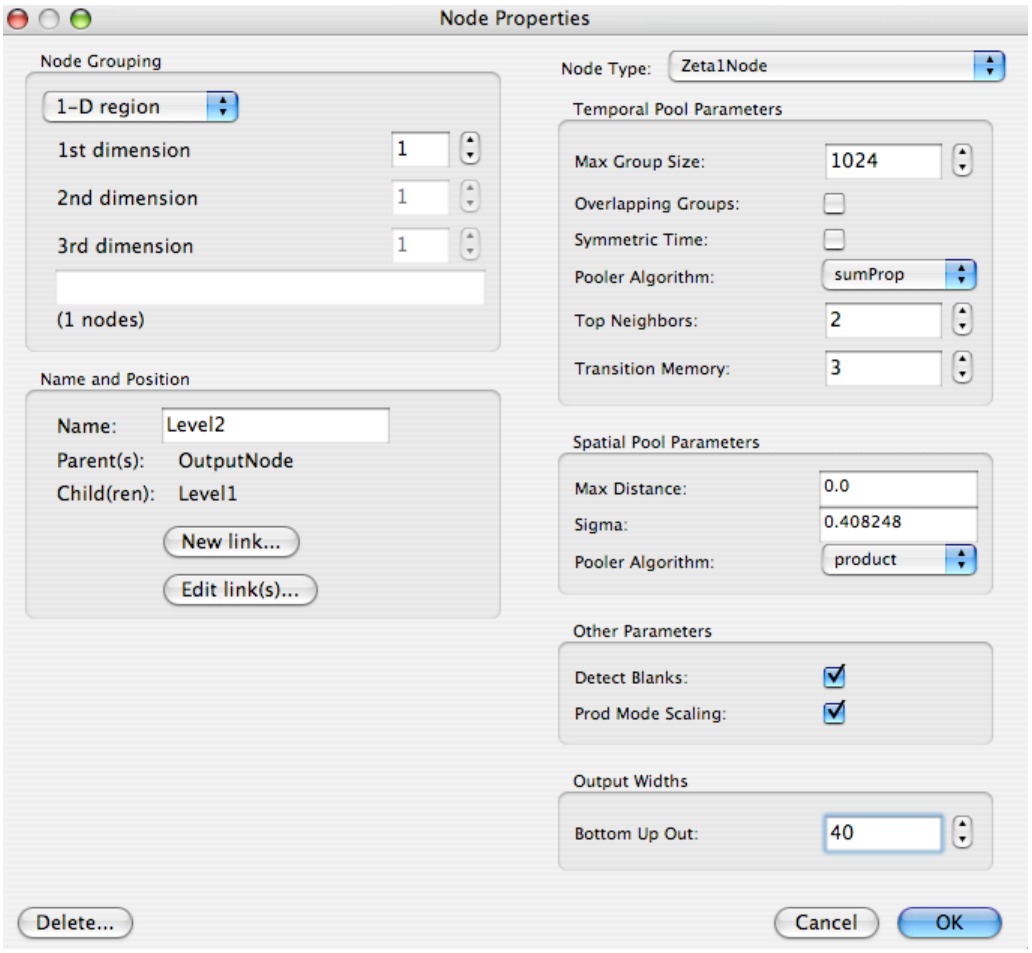


Figure 98: Top-Level Node Settings

B.11. Supervised Network Settings for Log-transformed $N = 2$

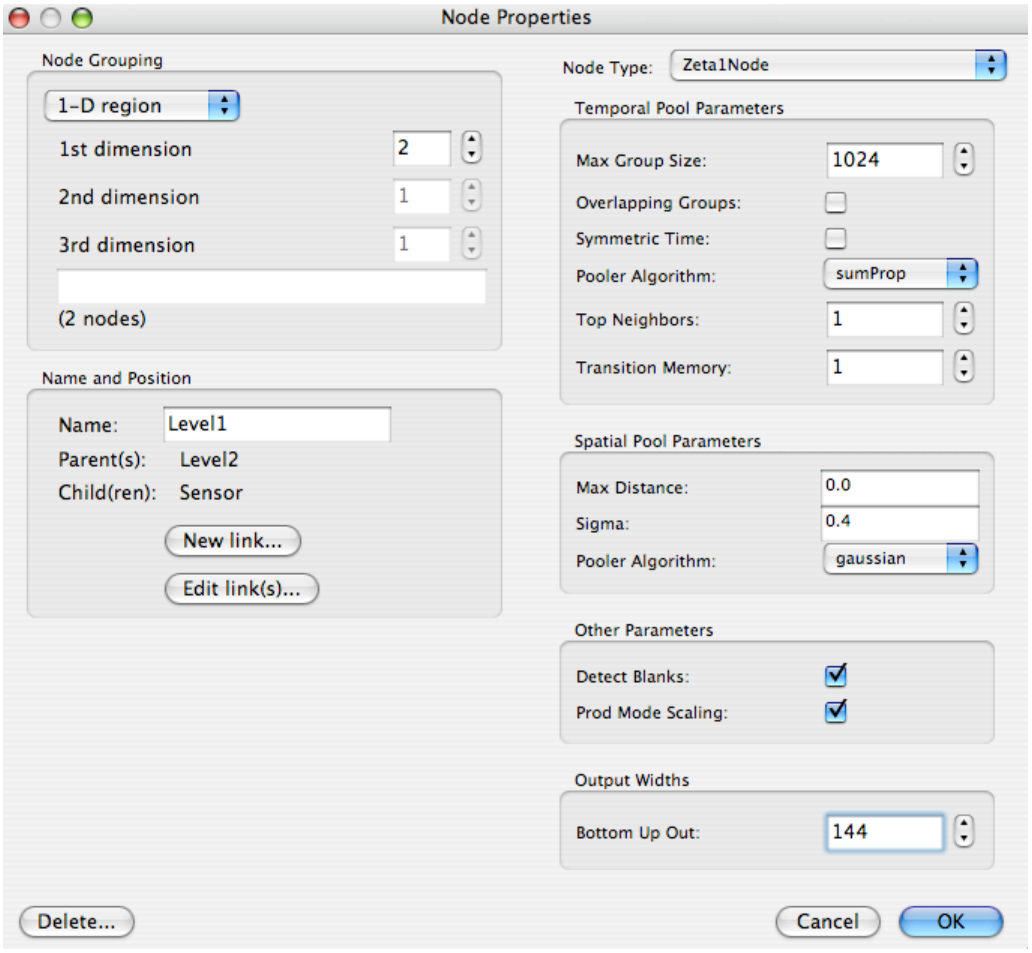


Figure 99: Bottom-Level Node Settings

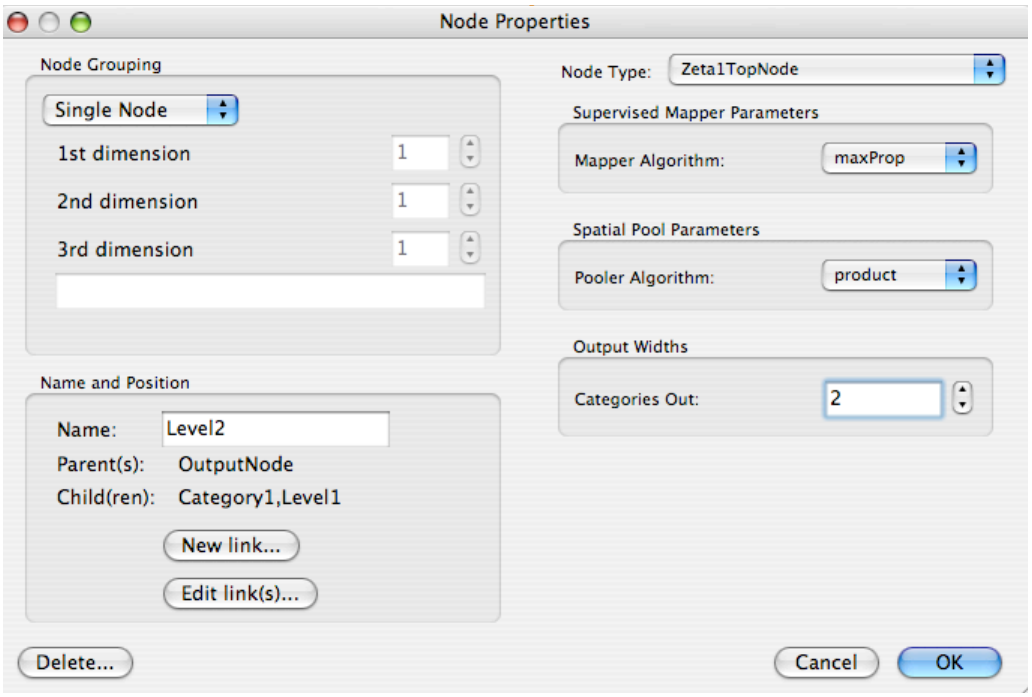


Figure 100: Top-Level Node Settings

B.12. Unsupervised Network Settings for Un-transformed $N = 4$

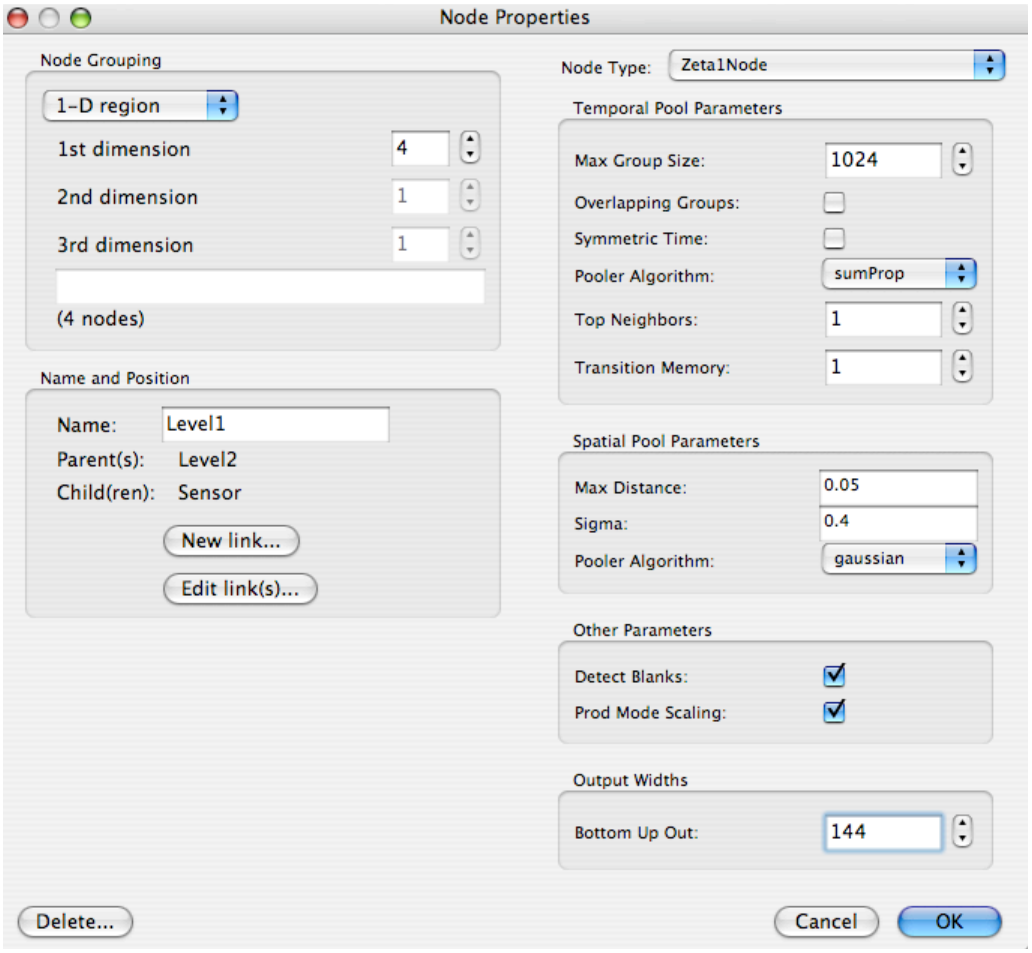


Figure 101: Bottom-Level Node Settings

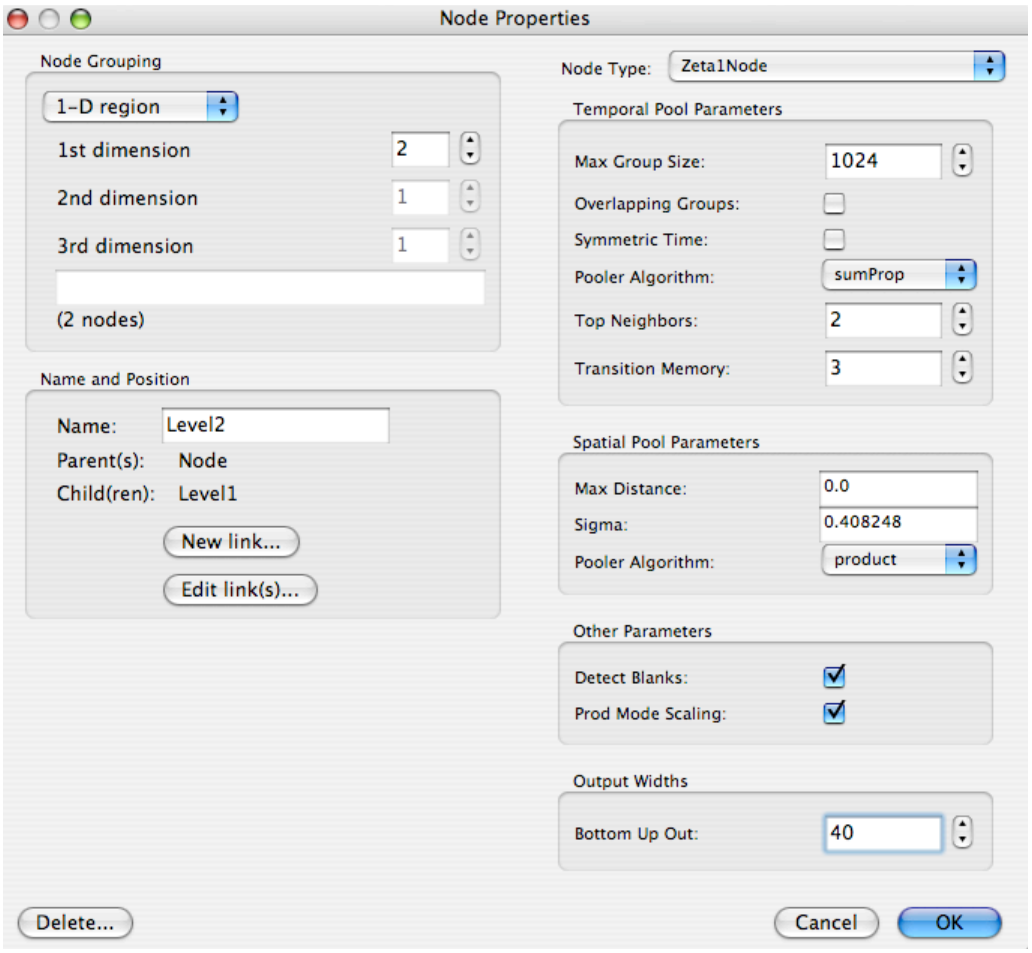


Figure 102: Second-Level Node Settings

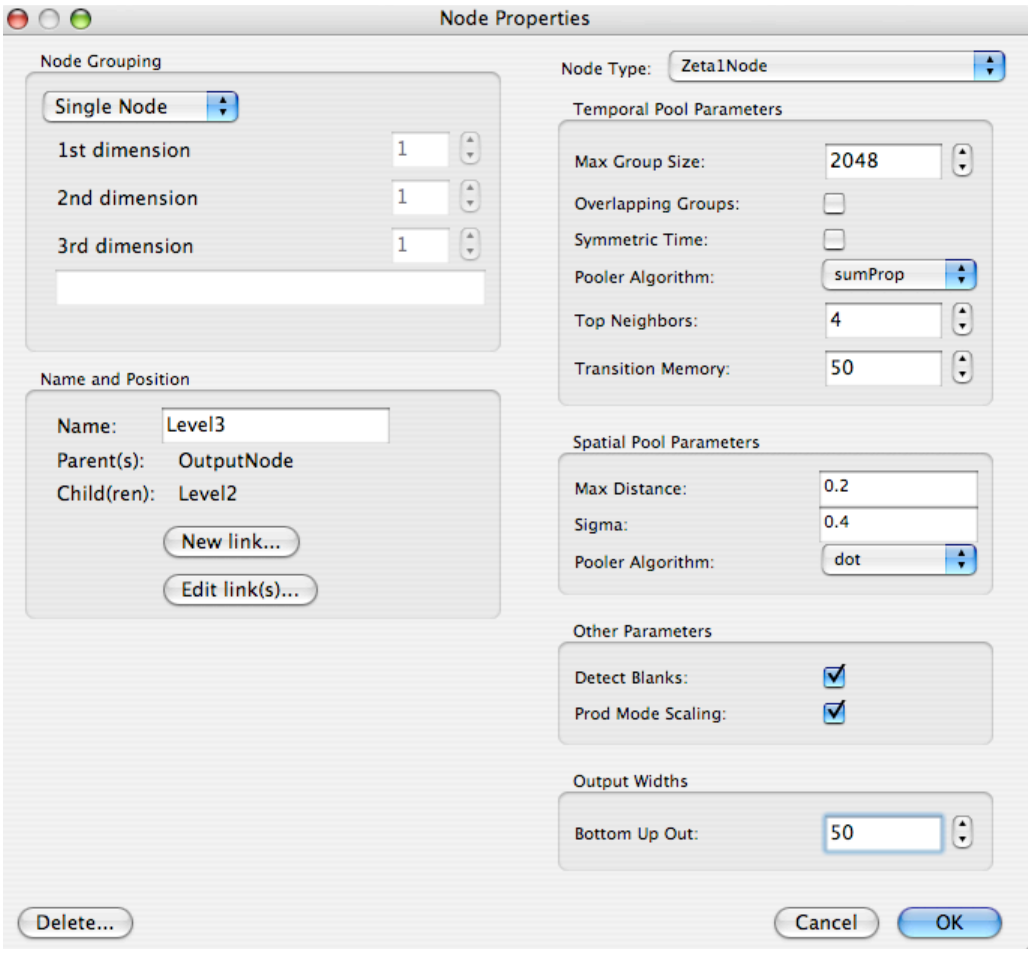


Figure 103: Top-Level Node Settings

B.13. Unsupervised Network Inference on $N = 4$ Un-transformed data

Time	First	Second
0	Group 0 0.190909	Group 1 0.009091
1	Group 0 1.809091	Group 1 0.090909
2	Group 0 1.809091	Group 1 0.090909
3	Group 0 1.809091	Group 1 0.090909
4	Group 0 1.809091	Group 1 0.090909
5	Group 0 1.809091	Group 1 0.090909
6	Group 0 1.809091	Group 1 0.090909
7	Group 0 1.809091	Group 1 0.090909
8	Group 0 1.809091	Group 1 0.090909
9	Group 0 1.809091	Group 1 0.090909
10	Group 0 0.909091	Group 1 0.190909
11	Group 1 1.8	Group 2 0.1
12	Group 1 1.8	Group 2 0.1
13	Group 1 1.8	Group 2 0.1
14	Group 1 1.8	Group 2 0.1
15	Group 1 1.8	Group 2 0.1
16	Group 1 1.8	Group 2 0.1
17	Group 1 1.8	Group 2 0.1
18	Group 1 1.8	Group 2 0.1

Time	First	Second
19	Group 1 1.8	Group 2 0.1
20	Group 1 0.9	Group 2 0.2
21	Group 2 1.8	Group 3 0.1
22	Group 2 1.8	Group 3 0.1
23	Group 2 1.8	Group 3 0.1
24	Group 2 1.8	Group 3 0.1
25	Group 2 1.8	Group 3 0.1
26	Group 2 1.8	Group 3 0.1
27	Group 2 1.8	Group 3 0.1
28	Group 2 1.8	Group 3 0.1
29	Group 2 1.8	Group 3 0.1
30	Group 2 0.9	Group 3 0.2
31	Group 3 1.8	Group 4 0.1
32	Group 3 1.8	Group 4 0.1
33	Group 3 1.8	Group 4 0.1
34	Group 3 1.8	Group 4 0.1
35	Group 3 1.8	Group 4 0.1
36	Group 3 1.8	Group 4 0.1
37	Group 3 1.8	Group 4 0.1

Time	First	Second
38	Group 3 1.8	Group 4 0.1
39	Group 3 1.8	Group 4 0.1
40	Group 3 0.9	Group 4 0.2
41	Group 4 1.8	Group 5 0.1
42	Group 4 1.8	Group 5 0.1
43	Group 4 1.8	Group 5 0.1
44	Group 4 1.8	Group 5 0.1
45	Group 4 1.8	Group 5 0.1
46	Group 4 1.8	Group 5 0.1
47	Group 4 1.8	Group 5 0.1
48	Group 4 1.8	Group 5 0.1
49	Group 4 1.8	Group 5 0.1
50	Group 4 0.9	Group 5 0.2
51	Group 5 1.8	Group 6 0.1
52	Group 5 1.8	Group 6 0.1
53	Group 5 1.8	Group 6 0.1
54	Group 5 1.8	Group 6 0.1
55	Group 5 1.8	Group 6 0.1
56	Group 5 1.8	Group 6 0.1

Time	First	Second
57	Group 5 1.8	Group 6 0.1
58	Group 5 1.8	Group 6 0.1
59	Group 5 1.8	Group 6 0.1
60	Group 5 0.9	Group 6 0.2
61	Group 6 1.8	Group 7 0.1
62	Group 6 1.8	Group 7 0.1
63	Group 6 1.8	Group 7 0.1
64	Group 6 1.8	Group 7 0.1
65	Group 6 1.8	Group 7 0.1
66	Group 6 1.8	Group 7 0.1
67	Group 6 1.8	Group 7 0.1
68	Group 6 1.8	Group 7 0.1
69	Group 6 1.8	Group 7 0.1
70	Group 6 0.9	Group 7 0.2
71	Group 7 1.8	Group 8 0.1
72	Group 7 1.8	Group 8 0.1
73	Group 7 1.8	Group 8 0.1
74	Group 7 1.8	Group 8 0.1
75	Group 7 1.8	Group 8 0.1

Time	First	Second
76	Group 7 1.8	Group 8 0.1
77	Group 7 1.8	Group 8 0.1
78	Group 7 1.8	Group 8 0.1
79	Group 7 1.8	Group 8 0.1
80	Group 7 0.9	Group 8 0.2
81	Group 8 1.8	Group 9 0.1
82	Group 8 1.8	Group 9 0.1
83	Group 8 1.8	Group 9 0.1
84	Group 8 1.8	Group 9 0.1
85	Group 8 1.8	Group 9 0.1
86	Group 8 1.8	Group 9 0.1
87	Group 8 1.8	Group 9 0.1
88	Group 8 1.8	Group 9 0.1
89	Group 8 1.8	Group 9 0.1
90	Group 8 0.9	Group 9 0.2
91	Group 9 1.8	Group 8 0
92	Group 9 1.8	Group 8 0
93	Group 9 1.8	Group 8 0
94	Group 9 1.8	Group 8 0
95	Group 9 1.8	Group 8 0
96	Group 9 1.8	Group 8 0
97	Group 9 1.8	Group 8 0
98	Group 9 1.8	Group 8 0
99	Group 9 1.8	Group 8 0

B.14. Supervised Network Settings for Un-transformed $N = 4$

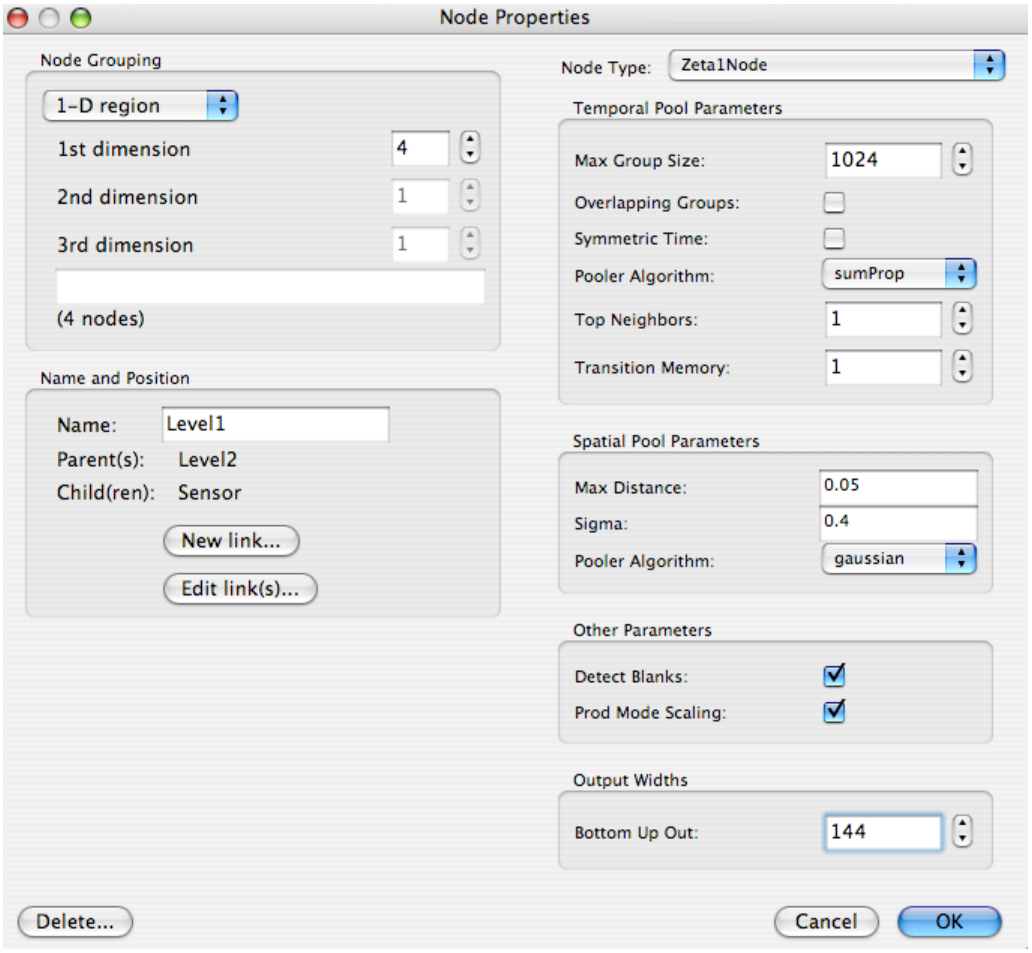


Figure 104: Bottom-Level Node Settings

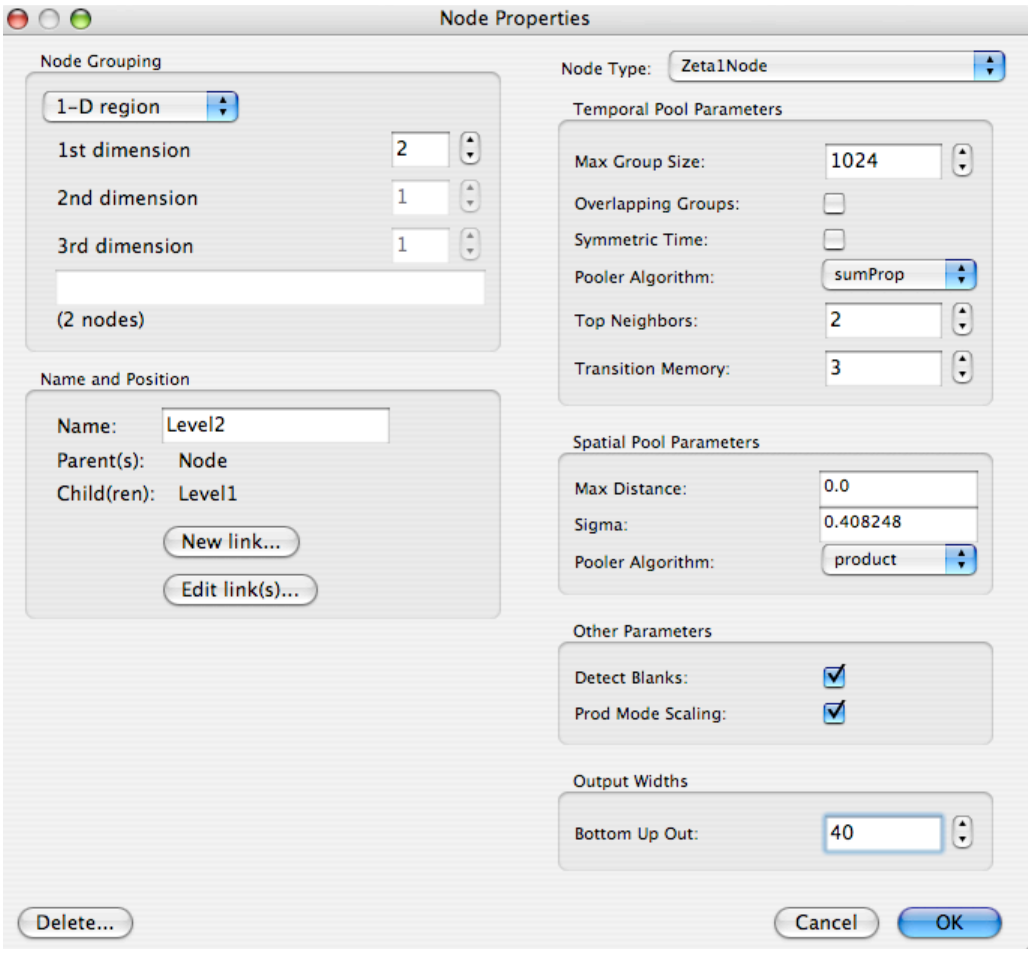


Figure 105: Second-Level Node Settings

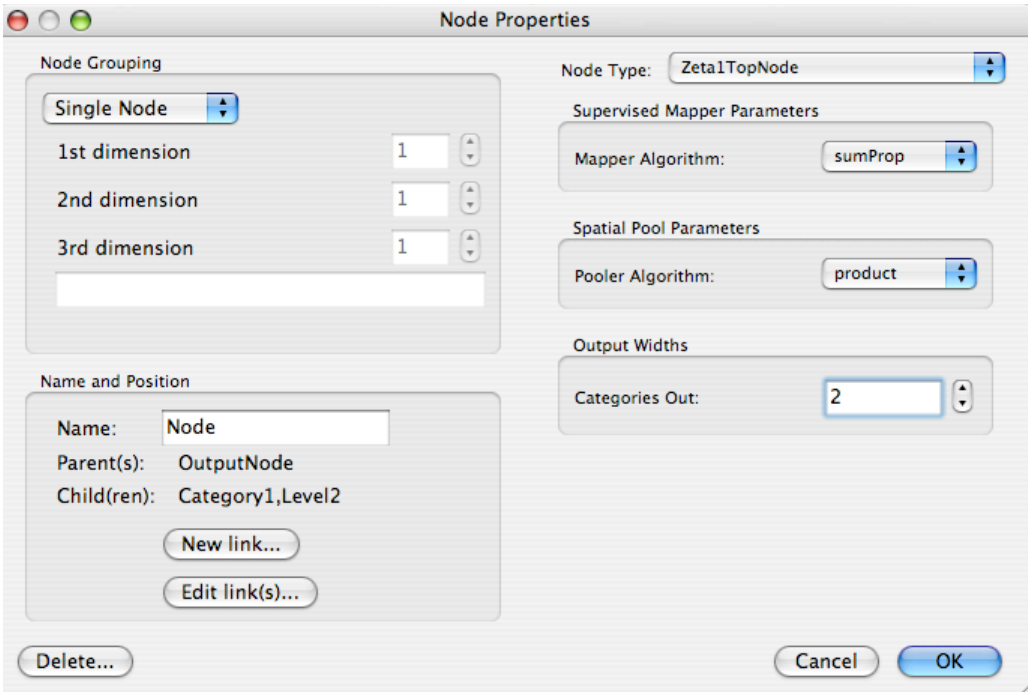


Figure 106: Top-Level Node Settings

B.15. *N = 4* Log-transformed Training Dataset

time	log(press)	log(dens)	log(h)	log(u)
time step	log(N/m^2)	log(kg/m^3)	log(J/kg/K)	log(m/s)
0	5.106530854	0.169674434	5.437583054	2.614797202
1	5.351069993	0.38102299	5.514115047	2.403448646
2	5.351069993	0.38102299	5.514115047	2.403448646
3	5.351069993	0.38102299	5.514115047	2.403448646
4	5.351069993	0.38102299	5.514115047	2.403448646
5	5.351069993	0.38102299	5.514115047	2.403448646
6	5.351069993	0.38102299	5.514115047	2.403448646
7	5.351069993	0.38102299	5.514115047	2.403448646
8	5.351069993	0.38102299	5.514115047	2.403448646
9	5.351069993	0.38102299	5.514115047	2.403448646
10	5.351069993	0.38102299	5.514115047	2.577350388
11	5.400338413	0.416201481	5.528204976	2.542171898
12	5.400338413	0.416201481	5.528204976	2.542171898
13	5.400338413	0.416201481	5.528204976	2.542171898
14	5.400338413	0.416201481	5.528204976	2.542171898
15	5.400338413	0.416201481	5.528204976	2.542171898
16	5.400338413	0.416201481	5.528204976	2.542171898
17	5.400338413	0.416201481	5.528204976	2.542171898
18	5.400338413	0.416201481	5.528204976	2.542171898
19	5.400338413	0.416201481	5.528204976	2.542171898
20	5.400338413	0.416201481	5.528204976	2.715591936
21	5.736010892	0.650205344	5.629873592	2.481588073
22	5.736010892	0.650205344	5.629873592	2.481588073
23	5.736010892	0.650205344	5.629873592	2.481588073
24	5.736010892	0.650205344	5.629873592	2.481588073
25	5.736010892	0.650205344	5.629873592	2.481588073
26	5.736010892	0.650205344	5.629873592	2.481588073
27	5.736010892	0.650205344	5.629873592	2.481588073
28	5.736010892	0.650205344	5.629873592	2.481588073
29	5.736010892	0.650205344	5.629873592	2.481588073
30	5.736010892	0.650205344	5.629873592	2.8860909
31	6.324944382	1.040754688	5.828257738	2.495541555
32	6.324944382	1.040754688	5.828257738	2.495541555
33	6.324944382	1.040754688	5.828257738	2.495541555
34	6.324944382	1.040754688	5.828257738	2.495541555
35	6.324944382	1.040754688	5.828257738	2.495541555
36	6.324944382	1.040754688	5.828257738	2.495541555
37	6.324944382	1.040754688	5.828257738	2.495541555
38	6.324944382	1.040754688	5.828257738	2.495541555
39	6.324944382	1.040754688	5.828257738	2.495541555
40	6.324944382	1.040754688	5.828257738	2.968128343
41	6.878032717	1.410722979	6.011377783	2.598160053
42	6.878032717	1.410722979	6.011377783	2.598160053
43	6.878032717	1.410722979	6.011377783	2.598160053
44	6.878032717	1.410722979	6.011377783	2.598160053
45	6.878032717	1.410722979	6.011377783	2.598160053
46	6.878032717	1.410722979	6.011377783	2.598160053
47	6.878032717	1.410722979	6.011377783	2.598160053
48	6.878032717	1.410722979	6.011377783	2.598160053
49	6.878032717	1.410722979	6.011377783	2.598160053
50	6.878032717	1.410722979	6.011377783	2.944558987
51	7.186462397	1.626549504	6.103980937	2.728732461
52	7.186462397	1.626549504	6.103980937	2.728732461
53	7.186462397	1.626549504	6.103980937	2.728732461
54	7.186462397	1.626549504	6.103980937	2.728732461
55	7.186462397	1.626549504	6.103980937	2.728732461
56	7.186462397	1.626549504	6.103980937	2.728732461
57	7.186462397	1.626549504	6.103980937	2.728732461
58	7.186462397	1.626549504	6.103980937	2.728732461
59	7.186462397	1.626549504	6.103980937	2.728732461

time	log(press)	log(dens)	log(h)	log(u)
time step	log(N/m^2)	log(kg/m^3)	log(J/kg/K)	log(m/s)
60	7.186462397	1.626549504	6.103980937	3.1457091
61	7.822361834	2.04317106	6.323258818	2.729087544
62	7.822361834	2.04317106	6.323258818	2.729087544
63	7.822361834	2.04317106	6.323258818	2.729087544
64	7.822361834	2.04317106	6.323258818	2.729087544
65	7.822361834	2.04317106	6.323258818	2.729087544
66	7.822361834	2.04317106	6.323258818	2.729087544
67	7.822361834	2.04317106	6.323258818	2.729087544
68	7.822361834	2.04317106	6.323258818	2.729087544
69	7.822361834	2.04317106	6.323258818	2.729087544
70	7.822361834	2.04317106	6.323258818	3.234836667
71	8.415577111	2.436139923	6.523505233	2.841867803
72	8.415577111	2.436139923	6.523505233	2.841867803
73	8.415577111	2.436139923	6.523505233	2.841867803
74	8.415577111	2.436139923	6.523505233	2.841867803
75	8.415577111	2.436139923	6.523505233	2.841867803
76	8.415577111	2.436139923	6.523505233	2.841867803
77	8.415577111	2.436139923	6.523505233	2.841867803
78	8.415577111	2.436139923	6.523505233	2.841867803
79	8.415577111	2.436139923	6.523505233	2.841867803
80	8.415577111	2.436139923	6.523505233	3.086543932
81	8.473714237	2.477641069	6.540141213	3.045042787
82	8.473714237	2.477641069	6.540141213	3.045042787
83	8.473714237	2.477641069	6.540141213	3.045042787
84	8.473714237	2.477641069	6.540141213	3.045042787
85	8.473714237	2.477641069	6.540141213	3.045042787
86	8.473714237	2.477641069	6.540141213	3.045042787
87	8.473714237	2.477641069	6.540141213	3.045042787
88	8.473714237	2.477641069	6.540141213	3.045042787
89	8.473714237	2.477641069	6.540141213	3.045042787
90	8.473714237	2.477641069	6.540141213	3.361036461
91	9.103894516	2.89114331	6.756819249	2.947534219
92	9.103894516	2.89114331	6.756819249	2.947534219
93	9.103894516	2.89114331	6.756819249	2.947534219
94	9.103894516	2.89114331	6.756819249	2.947534219
95	9.103894516	2.89114331	6.756819249	2.947534219
96	9.103894516	2.89114331	6.756819249	2.947534219
97	9.103894516	2.89114331	6.756819249	2.947534219
98	9.103894516	2.89114331	6.756819249	2.947534219
99	9.103894516	2.89114331	6.756819249	2.947534219

B.16. Unsupervised Two-Level Network Settings for Log-transformed *N* = *4*

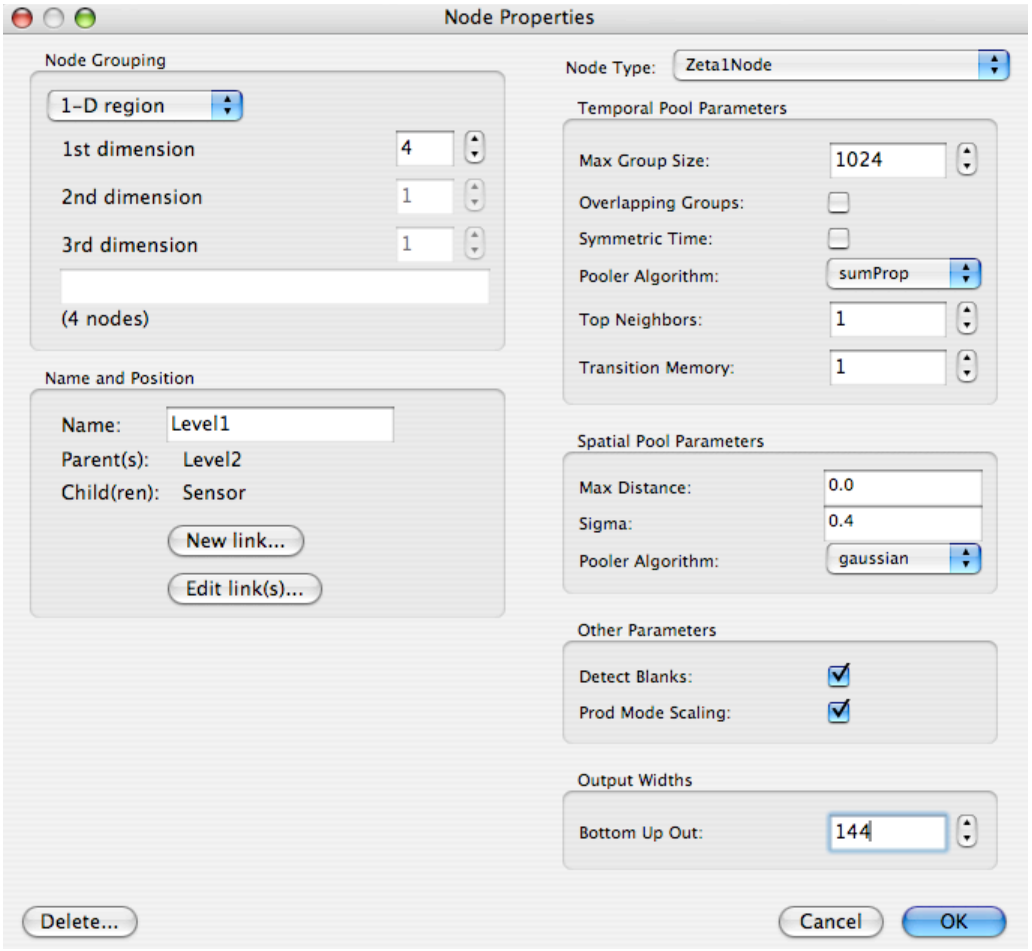


Figure 107: Bottom-Level Node Settings

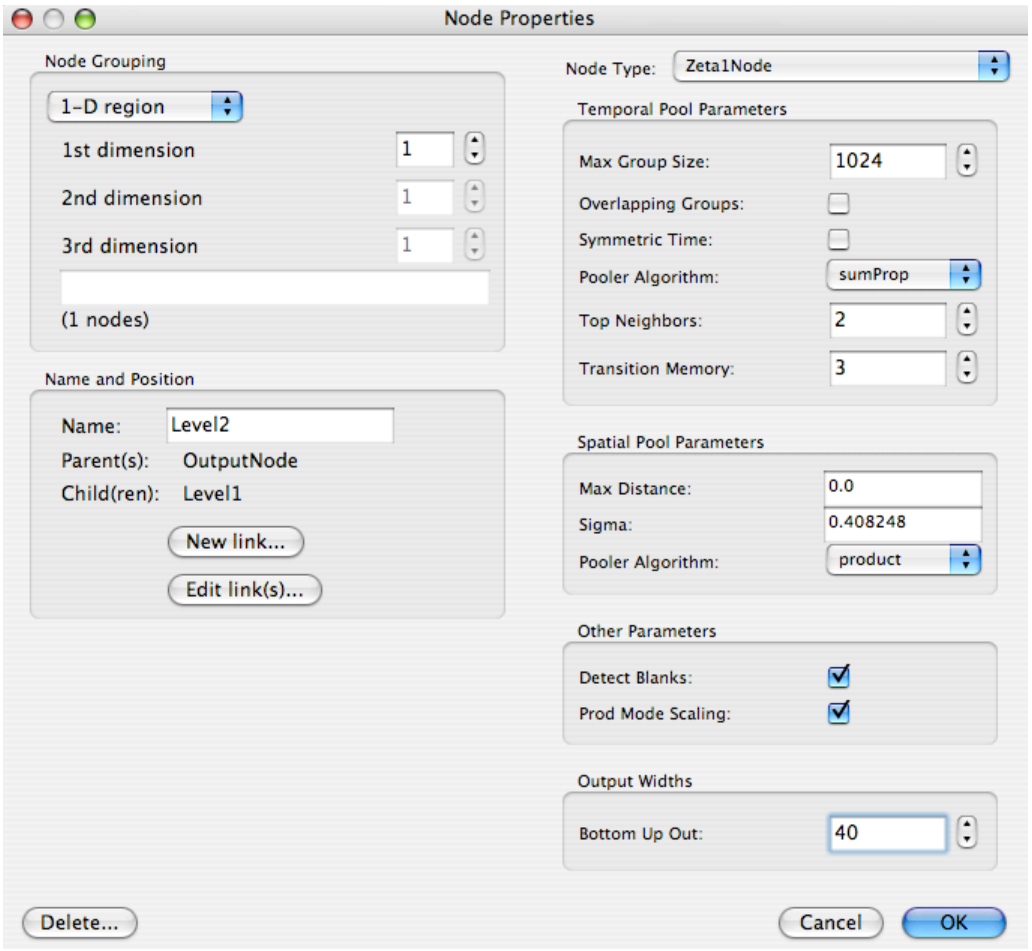


Figure 108: Top-Level Node Settings

B.17. *N = 4* Testing: Log-transformed 5-10% Perturbation (Constant *T*) of Training Dataset

time	log(press)	log(dens)	log(h)	log(u)
time step	log(N/m^2)	log(kg/m^3)	log(J/kg/K)	log(m/s)
0	5.115509185	0.221994176	5.437583054	2.61814713
1	5.32931537	0.359268367	5.514115047	2.424124016
2	5.348450171	0.378403168	5.514115047	2.40209657
3	5.330912099	0.360865097	5.514115047	2.393114689
4	5.340574955	0.370527952	5.514115047	2.401255809
5	5.343250972	0.373203969	5.514115047	2.404912308
6	5.367721774	0.397674771	5.514115047	2.413564276
7	5.333364961	0.363317958	5.514115047	2.388677879
8	5.331704896	0.361657894	5.514115047	2.389691413
9	5.347740514	0.377693511	5.514115047	2.388260835
10	5.342629397	0.372582394	5.514115047	2.592663969
11	5.387300129	0.403163197	5.528204976	2.546231628
12	5.379853486	0.395716554	5.528204976	2.536944168
13	5.396329618	0.412192686	5.528204976	2.556721664
14	5.403812929	0.419675997	5.528204976	2.548122554
15	5.395823128	0.411686196	5.528204976	2.556326972
16	5.382595377	0.398458445	5.528204976	2.557317514
17	5.392324408	0.408187476	5.528204976	2.545692125
18	5.419990312	0.43585338	5.528204976	2.544419863
19	5.389675189	0.405538256	5.528204976	2.536054298
20	5.418714485	0.434577553	5.528204976	2.696210663
21	5.73757555	0.651770002	5.629873592	2.47488982
22	5.747018047	0.661212499	5.629873592	2.467400082
23	5.739945737	0.654140189	5.629873592	2.49212799
24	5.725847552	0.640042004	5.629873592	2.501126386
25	5.725483065	0.639677517	5.629873592	2.465284035
26	5.732790042	0.646984494	5.629873592	2.488091421
27	5.749095075	0.663289527	5.629873592	2.462362645
28	5.735636378	0.64983083	5.629873592	2.480652758
29	5.748319591	0.662514043	5.629873592	2.477548859
30	5.75001572	0.664210172	5.629873592	2.88831737
31	6.341168696	1.056979003	5.828257738	2.50350566
32	6.325437122	1.041247428	5.828257738	2.481951385
33	6.326699418	1.042509725	5.828257738	2.485956244
34	6.315058041	1.030868347	5.828257738	2.486373391
35	6.304696996	1.020507302	5.828257738	2.480936586
36	6.306759633	1.02256994	5.828257738	2.494426979
37	6.319773605	1.035583912	5.828257738	2.495092336
38	6.311659398	1.027469705	5.828257738	2.502548706
39	6.32675425	1.042564556	5.828257738	2.490910139
40	6.312740042	1.028550349	5.828257738	2.986045642
41	6.865627603	1.398317865	6.011377783	2.597924527
42	6.881583644	1.414273906	6.011377783	2.618116668
43	6.882866297	1.415556559	6.011377783	2.578862727
44	6.855944161	1.388634423	6.011377783	2.596297426
45	6.865954008	1.39864427	6.011377783	2.582903336
46	6.878135094	1.410825356	6.011377783	2.580253948
47	6.877116753	1.409807015	6.011377783	2.612042925
48	6.886366368	1.41905663	6.011377783	2.615432691
49	6.884681768	1.41737203	6.011377783	2.598238498
50	6.870770126	1.403460388	6.011377783	2.944583961
51	7.194798969	1.634886076	6.103980937	2.707221783
52	7.166261744	1.606348851	6.103980937	2.730560818
53	7.165353623	1.60544073	6.103980937	2.744474096
54	7.201076089	1.641163196	6.103980937	2.735343229
55	7.200593232	1.640680339	6.103980937	2.722361769
56	7.197916516	1.638003623	6.103980937	2.739809733
57	7.196715298	1.636802406	6.103980937	2.722588781
58	7.173210979	1.613298086	6.103980937	2.722826572
59	7.181506813	1.62159392	6.103980937	2.738837452

time	log(press)	log(dens)	log(h)	log(u)
time step	log(N/m^2)	log(kg/m^3)	log(J/kg/K)	log(m/s)
60	7.197062523	1.63714963	6.103980937	3.125763585
61	7.829700233	2.050509458	6.323258818	2.744943013
62	7.80444986	2.025259085	6.323258818	2.713149943
63	7.829584499	2.050393724	6.323258818	2.741450346
64	7.824542105	2.045351331	6.323258818	2.714720432
65	7.801161702	2.021970928	6.323258818	2.748787555
66	7.81388001	2.034689235	6.323258818	2.711766497
67	7.836922903	2.057732129	6.323258818	2.73450495
68	7.843530961	2.064340187	6.323258818	2.742739917
69	7.842701539	2.063510765	6.323258818	2.708959577
70	7.825019905	2.045829131	6.323258818	3.245903525
71	8.405177764	2.425740576	6.523505233	2.851129736
72	8.411766704	2.432329516	6.523505233	2.845946504
73	8.410285877	2.430848689	6.523505233	2.835294052
74	8.40983286	2.430395672	6.523505233	2.840898774
75	8.413423726	2.433986538	6.523505233	2.829049132
76	8.411118	2.431680812	6.523505233	2.842593783
77	8.424444974	2.445007786	6.523505233	2.854269919
78	8.428855362	2.449418174	6.523505233	2.828538261
79	8.409959151	2.430521962	6.523505233	2.823260436
80	8.430431817	2.450994629	6.523505233	3.094734674
81	8.461490367	2.465417198	6.540141213	3.046042488
82	8.461318798	2.465245629	6.540141213	3.04991063
83	8.45351621	2.457443041	6.540141213	3.065941625
84	8.481812727	2.485739558	6.540141213	3.058860971
85	8.474262226	2.478189058	6.540141213	3.063178489
86	8.476500141	2.480426973	6.540141213	3.055953175
87	8.472747101	2.476673933	6.540141213	3.022845136
88	8.455548549	2.459475381	6.540141213	3.026064827
89	8.489171773	2.493098604	6.540141213	3.037992408
90	8.479731281	2.483658113	6.540141213	3.356960902
91	9.120241148	2.907489943	6.756819249	2.949882927
92	9.119838593	2.907087388	6.756819249	2.963457031
93	9.082095662	2.869344456	6.756819249	2.94301656
94	9.114185151	2.901433946	6.756819249	2.965396099
95	9.09707053	2.884319325	6.756819249	2.95007211
96	9.117591514	2.904840309	6.756819249	2.950859138
97	9.115979435	2.90322823	6.756819249	2.957101678
98	9.123203019	2.910451814	6.756819249	2.925607653
99	9.123504065	2.910752859	6.756819249	2.952944148

B.18. Unsupervised Network Inference on $N = 4$ Log-transformed Training Data

Time	First	Second
0	Group 0 0.65449	Group 1 0.615408
1	Group 0 0.961609	Group 1 0.928283
2	Group 0 0.961609	Group 1 0.928283
3	Group 0 0.961609	Group 1 0.928283
4	Group 0 0.961609	Group 1 0.928283
5	Group 0 0.961609	Group 1 0.928283
6	Group 0 0.961609	Group 1 0.928283
7	Group 0 0.961609	Group 1 0.928283
8	Group 0 0.961609	Group 1 0.928283
9	Group 0 0.961609	Group 1 0.928283
10	Group 1 0.985765	Group 0 0.88937
11	Group 1 0.998418	Group 0 0.897843
12	Group 1 0.998418	Group 0 0.897843
13	Group 1 0.998418	Group 0 0.897843
14	Group 1 0.998418	Group 0 0.897843
15	Group 1 0.998418	Group 0 0.897843
16	Group 1 0.998418	Group 0 0.897843
17	Group 1 0.998418	Group 0 0.897843
18	Group 1 0.998418	Group 0 0.897843
19	Group 1 0.998418	Group 0 0.897843
20	Group 1 0.912339	Group 0 0.715417
21	Group 2 0.948353	Group 1 0.557266
22	Group 2 0.948353	Group 1 0.557266
23	Group 2 0.948353	Group 1 0.557266
24	Group 2 0.948353	Group 1 0.557266
25	Group 2 0.948353	Group 1 0.557266
26	Group 2 0.948353	Group 1 0.557266
27	Group 2 0.948353	Group 1 0.557266
28	Group 2 0.948353	Group 1 0.557266
29	Group 2 0.948353	Group 1 0.557266
30	Group 2 0.592129	Group 1 0.392548
31	Group 3 0.91153	Group 4 0.246329
32	Group 3 0.91153	Group 4 0.246329
33	Group 3 0.91153	Group 4 0.246329
34	Group 3 0.91153	Group 4 0.246329
35	Group 3 0.91153	Group 4 0.246329
36	Group 3 0.91153	Group 4 0.246329
37	Group 3 0.91153	Group 4 0.246329
38	Group 3 0.91153	Group 4 0.246329
39	Group 3 0.91153	Group 4 0.246329
40	Group 3 0.466035	Group 4 0.232446
41	Group 4 0.914716	Group 5 0.602242
42	Group 4 0.914716	Group 5 0.602242
43	Group 4 0.914716	Group 5 0.602242
44	Group 4 0.914716	Group 5 0.602242
45	Group 4 0.914716	Group 5 0.602242
46	Group 4 0.914716	Group 5 0.602242
47	Group 4 0.914716	Group 5 0.602242
48	Group 4 0.914716	Group 5 0.602242
49	Group 4 0.914716	Group 5 0.602242
50	Group 4 0.641107	Group 5 0.586476
51	Group 5 0.954052	Group 4 0.53572
52	Group 5 0.954052	Group 4 0.53572
53	Group 5 0.954052	Group 4 0.53572
54	Group 5 0.954052	Group 4 0.53572
55	Group 5 0.954052	Group 4 0.53572
56	Group 5 0.954052	Group 4 0.53572
57	Group 5 0.954052	Group 4 0.53572
58	Group 5 0.954052	Group 4 0.53572
59	Group 5 0.954052	Group 4 0.53572
60	Group 5 0.577815	Group 4 0.222891
61	Group 6 0.908219	Group 7 0.201775
62	Group 6 0.908219	Group 7 0.201775
63	Group 6 0.908219	Group 7 0.201775
64	Group 6 0.908219	Group 7 0.201775
65	Group 6 0.908219	Group 7 0.201775
66	Group 6 0.908219	Group 7 0.201775
67	Group 6 0.908219	Group 7 0.201775
68	Group 6 0.908219	Group 7 0.201775
69	Group 6 0.908219	Group 7 0.201775
70	Group 6 0.418005	Group 7 0.200708
71	Group 7 0.91119	Group 8 0.860827
72	Group 7 0.91119	Group 8 0.860827
73	Group 7 0.91119	Group 8 0.860827
74	Group 7 0.91119	Group 8 0.860827
75	Group 7 0.91119	Group 8 0.860827
76	Group 7 0.91119	Group 8 0.860827
77	Group 7 0.91119	Group 8 0.860827
78	Group 7 0.91119	Group 8 0.860827
79	Group 7 0.91119	Group 8 0.860827
80	Group 8 0.980243	Group 7 0.763367
81	Group 8 0.997805	Group 7 0.789247
82	Group 8 0.997805	Group 7 0.789247
83	Group 8 0.997805	Group 7 0.789247
84	Group 8 0.997805	Group 7 0.789247
85	Group 8 0.997805	Group 7 0.789247
86	Group 8 0.997805	Group 7 0.789247
87	Group 8 0.997805	Group 7 0.789247
88	Group 8 0.997805	Group 7 0.789247
89	Group 8 0.997805	Group 7 0.789247
90	Group 8 0.736459	Group 7 0.393279
91	Group 9 0.908574	Group 8 0.137277
92	Group 9 0.908574	Group 8 0.137277
93	Group 9 0.908574	Group 8 0.137277
94	Group 9 0.908574	Group 8 0.137277
95	Group 9 0.908574	Group 8 0.137277
96	Group 9 0.908574	Group 8 0.137277
97	Group 9 0.908574	Group 8 0.137277
98	Group 9 0.908574	Group 8 0.137277
99	Group 9 0.908574	Group 8 0.137277

B.19. Unsupervised Network Inference on Log-transformed 5-10% (Constant *T)* Perturbation *N = 4*

Time	First	Second
0	Group 0 0.699715	Group 1 0.676088
1	Group 0 0.967364	Group 1 0.935762
2	Group 0 0.96196	Group 1 0.926016
3	Group 0 0.964367	Group 1 0.911019
4	Group 0 0.963296	Group 1 0.921861
5	Group 0 0.963119	Group 1 0.926037
6	Group 0 0.959545	Group 1 0.94412
7	Group 0 0.963545	Group 1 0.90849
8	Group 0 0.963932	Group 1 0.908594
9	Group 0 0.960961	Group 1 0.914391
10	Group 1 0.978865	Group 0 0.877831
11	Group 1 0.999187	Group 0 0.903506
12	Group 1 0.999374	Group 0 0.914975
13	Group 1 0.998943	Group 0 0.890328
14	Group 1 0.998367	Group 0 0.891465
15	Group 1 0.998961	Group 0 0.890933
16	Group 1 0.999676	Group 0 0.898879
17	Group 1 0.998913	Group 0 0.900595
18	Group 1 0.997451	Group 0 0.883761
19	Group 1 0.998841	Group 0 0.909097
20	Group 1 0.93039	Group 0 0.731216
21	Group 2 0.947616	Group 1 0.552662
22	Group 2 0.94554	Group 1 0.532581
23	Group 2 0.94842	Group 1 0.551718
24	Group 2 0.951587	Group 1 0.582561
25	Group 2 0.949017	Group 1 0.575088
26	Group 2 0.949378	Group 1 0.565233
27	Group 2 0.944872	Group 1 0.52755
28	Group 2 0.948351	Group 1 0.557813
29	Group 2 0.946007	Group 1 0.532229
30	Group 2 0.586433	Group 1 0.371437
31	Group 3 0.910645	Group 4 0.267907
32	Group 3 0.911121	Group 4 0.24322
33	Group 3 0.911143	Group 4 0.245723
34	Group 3 0.911979	Group 4 0.233015
35	Group 3 0.912595	Group 4 0.220927
36	Group 3 0.912853	Group 4 0.226462
37	Group 3 0.911888	Group 4 0.240431
38	Group 3 0.912723	Group 4 0.233688
39	Group 3 0.911275	Group 4 0.247127
40	Group 3 0.444182	Group 4 0.218432
41	Group 4 0.915799	Group 5 0.580858
42	Group 4 0.915098	Group 5 0.620382
43	Group 4 0.913686	Group 5 0.599474
44	Group 4 0.916644	Group 5 0.563961
45	Group 4 0.915231	Group 5 0.572956
46	Group 4 0.91411	Group 5 0.592062
47	Group 4 0.915276	Group 5 0.608801
48	Group 4 0.914596	Group 5 0.627409
49	Group 4 0.914165	Group 5 0.614045
50	Group 4 0.642038	Group 5 0.575023
51	Group 5 0.951092	Group 4 0.530312
52	Group 5 0.957894	Group 4 0.571833
53	Group 5 0.959167	Group 4 0.567164
54	Group 5 0.951987	Group 4 0.507786
55	Group 5 0.951165	Group 4 0.513952
56	Group 5 0.952844	Group 4 0.511254
57	Group 5 0.951836	Group 4 0.520479
58	Group 5 0.956003	Group 4 0.56235
59	Group 5 0.955691	Group 4 0.540135
60	Group 5 0.605176	Group 4 0.230237
61	Group 6 0.908162	Group 7 0.213193
62	Group 6 0.90887	Group 7 0.1816
63	Group 6 0.908094	Group 7 0.212146
64	Group 6 0.907804	Group 7 0.200291
65	Group 6 0.909946	Group 7 0.187388
66	Group 6 0.908307	Group 7 0.189579
67	Group 6 0.907574	Group 7 0.217936
68	Group 6 0.907409	Group 7 0.227347
69	Group 6 0.906822	Group 7 0.217432
70	Group 6 0.403777	Group 7 0.199626
71	Group 7 0.912205	Group 8 0.866122
72	Group 7 0.911571	Group 8 0.86353
73	Group 7 0.911375	Group 8 0.851005
74	Group 7 0.911565	Group 8 0.856965
75	Group 7 0.910988	Group 8 0.845647
76	Group 7 0.911522	Group 8 0.859479
77	Group 7 0.910922	Group 8 0.879135
78	Group 7 0.909978	Group 8 0.852441
79	Group 7 0.911066	Group 8 0.837675
80	Group 8 0.986527	Group 7 0.752712
81	Group 8 0.998579	Group 7 0.795206
82	Group 8 0.998687	Group 7 0.791513
83	Group 8 0.999579	Group 7 0.780441
84	Group 8 0.997663	Group 7 0.77145
85	Group 8 0.998233	Group 7 0.771487
86	Group 8 0.997913	Group 7 0.777154
87	Group 8 0.997303	Group 7 0.811749
88	Group 8 0.998432	Group 7 0.818493
89	Group 8 0.99669	Group 7 0.787498
90	Group 8 0.742034	Group 7 0.396469
91	Group 9 0.907754	Group 8 0.123489
92	Group 9 0.908052	Group 8 0.124874
93	Group 9 0.909769	Group 8 0.157953
94	Group 9 0.908396	Group 8 0.129754
95	Group 9 0.909023	Group 8 0.143796
96	Group 9 0.907909	Group 8 0.125735
97	Group 9 0.908123	Group 8 0.12757
98	Group 9 0.907143	Group 8 0.11928
99	Group 9 0.907651	Group 8 0.121102

B.20. Unsupervised Network Inference on Log-transformed 5-10% (Non-constant *T*) Perturbation *N* = *4* Dataset

Time	First	Second
0	Group 0 0.639528	Group 1 0.598747
1	Group 0 0.967861	Group 1 0.934582
2	Group 0 0.962919	Group 1 0.92398
3	Group 0 0.963246	Group 1 0.913568
4	Group 0 0.963658	Group 1 0.921035
5	Group 0 0.963203	Group 1 0.925967
6	Group 0 0.96022	Group 1 0.942426
7	Group 0 0.962056	Group 1 0.911816
8	Group 0 0.960933	Group 1 0.915293
9	Group 0 0.961823	Group 1 0.912394
10	Group 1 0.97796	Group 0 0.878262
11	Group 1 0.998801	Group 0 0.898601
12	Group 1 0.99927	Group 0 0.913632
13	Group 1 0.998363	Group 0 0.883037
14	Group 1 0.998171	Group 0 0.888919
15	Group 1 0.998798	Group 0 0.888808
16	Group 1 0.999486	Group 0 0.896402
17	Group 1 0.9993	Group 0 0.90556
18	Group 1 0.998207	Group 0 0.893289
19	Group 1 0.998483	Group 0 0.904496
20	Group 1 0.930285	Group 0 0.730055
21	Group 2 0.94791	Group 1 0.556123
22	Group 2 0.944372	Group 1 0.518764
23	Group 2 0.946922	Group 1 0.534437
24	Group 2 0.948872	Group 1 0.551532
25	Group 2 0.948848	Group 1 0.573082
26	Group 2 0.948374	Group 1 0.553602
27	Group 2 0.945078	Group 1 0.530008
28	Group 2 0.948681	Group 1 0.561674
29	Group 2 0.946347	Group 1 0.536218
30	Group 2 0.586643	Group 1 0.373034
31	Group 3 0.91083	Group 4 0.264451
32	Group 3 0.911302	Group 4 0.240268
33	Group 3 0.911073	Group 4 0.24685
34	Group 3 0.912304	Group 4 0.228485
35	Group 3 0.911344	Group 4 0.238905
36	Group 3 0.912051	Group 4 0.237642
37	Group 3 0.91151	Group 4 0.246403
38	Group 3 0.912854	Group 4 0.231966
39	Group 3 0.911081	Group 4 0.250311
40	Group 3 0.444074	Group 4 0.21911
41	Group 4 0.915023	Group 5 0.596749
42	Group 4 0.914549	Group 5 0.631565
43	Group 4 0.913822	Group 5 0.596065
44	Group 4 0.915576	Group 5 0.583557
45	Group 4 0.915257	Group 5 0.572562
46	Group 4 0.914467	Group 5 0.583703
47	Group 4 0.915164	Group 5 0.610531
48	Group 4 0.914372	Group 5 0.631734
49	Group 4 0.914191	Group 5 0.612659
50	Group 4 0.640811	Group 5 0.590267
51	Group 5 0.949879	Group 4 0.517644
52	Group 5 0.954643	Group 4 0.539477
53	Group 5 0.958535	Group 4 0.56105
54	Group 5 0.953693	Group 4 0.524569
55	Group 5 0.953145	Group 4 0.53398
56	Group 5 0.954066	Group 4 0.523154
57	Group 5 0.953424	Group 4 0.536539
58	Group 5 0.953785	Group 4 0.539924
59	Group 5 0.954814	Group 4 0.531571
60	Group 5 0.606577	Group 4 0.236254
61	Group 6 0.908121	Group 7 0.21397
62	Group 6 0.908962	Group 7 0.180275
63	Group 6 0.908829	Group 7 0.199215
64	Group 6 0.90797	Group 7 0.197266
65	Group 6 0.909371	Group 7 0.195516
66	Group 6 0.908268	Group 7 0.190196
67	Group 6 0.907771	Group 7 0.213915
68	Group 6 0.908217	Group 7 0.210596
69	Group 6 0.907113	Group 7 0.210624
70	Group 6 0.404088	Group 7 0.197466
71	Group 7 0.911471	Group 8 0.870932
72	Group 7 0.911331	Group 8 0.865552
73	Group 7 0.911417	Group 8 0.850304
74	Group 7 0.911124	Group 8 0.860389
75	Group 7 0.911382	Group 8 0.843168
76	Group 7 0.910821	Group 8 0.864514
77	Group 7 0.911364	Group 8 0.875561
78	Group 7 0.909989	Group 8 0.852275
79	Group 7 0.910987	Group 8 0.83822
80	Group 8 0.980622	Group 7 0.753811
81	Group 8 0.997986	Group 7 0.789712
82	Group 8 0.997763	Group 7 0.782996
83	Group 8 0.999345	Group 7 0.778314
84	Group 8 0.99722	Group 7 0.76744
85	Group 8 0.998663	Group 7 0.775365
86	Group 8 0.997506	Group 7 0.773455
87	Group 8 0.997594	Group 7 0.814564
88	Group 8 0.997651	Group 7 0.811084
89	Group 8 0.997027	Group 7 0.790525
90	Group 8 0.742016	Group 7 0.396348
91	Group 9 0.907808	Group 8 0.124363
92	Group 9 0.908386	Group 8 0.130099
93	Group 9 0.909127	Group 8 0.147502
94	Group 9 0.909107	Group 8 0.14082
95	Group 9 0.909032	Group 8 0.143943
96	Group 9 0.907711	Group 8 0.122568
97	Group 9 0.908164	Group 8 0.128212
98	Group 9 0.907234	Group 8 0.120823
99	Group 9 0.907896	Group 8 0.125005

B.21. Supervised Two-Level Network Settings for Log-transformed $N = 4$

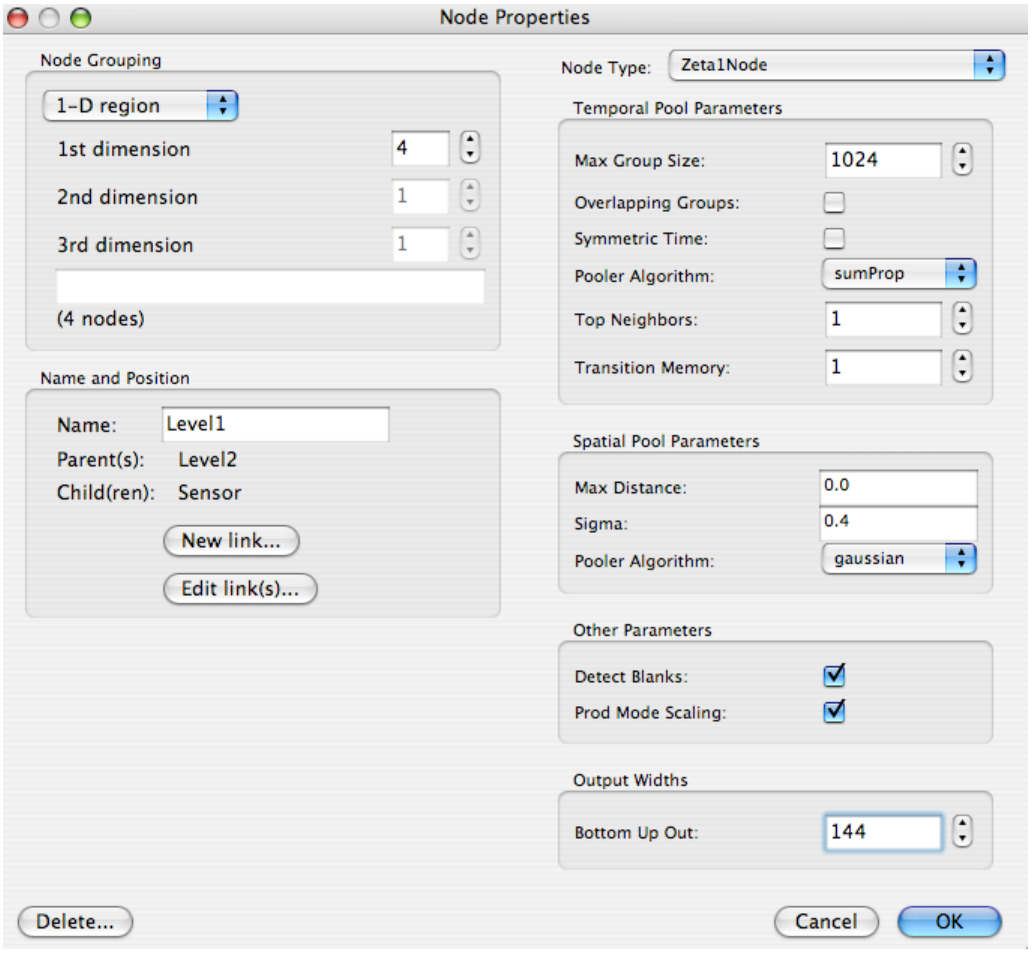


Figure 109: Bottom-Level Node Settings

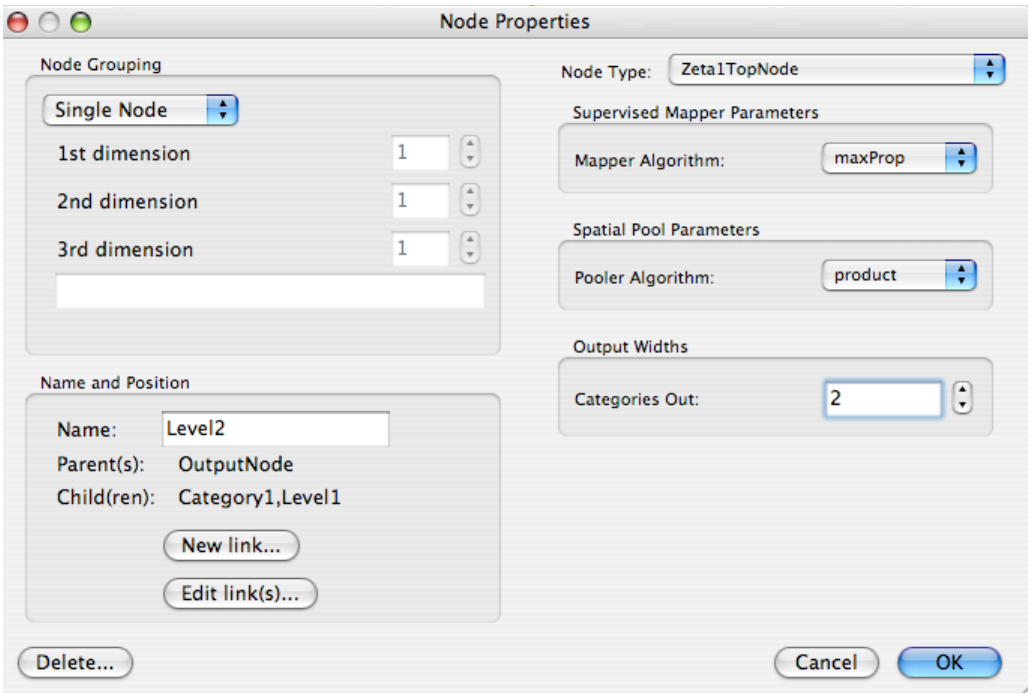


Figure 110: Top-Level Node Settings

APPENDIX C

IRAQ CONTEXT DATA

This appendix contains data pertinent to the Iraq context experiments. Each section title indicates the contents. In this appendix, the following will be found: training and testing data, network parameters and network inference results. The following data has been used either to train or test HTM networks used in developing the SA of the Iraq context. Similarly, the network parameters that follow have also been used to create SA of the Iraq context. The inference results below document a large amount of this recognition capability.

The title of each section indicates the characteristic information to be found in it. These sections are labeled and numbered to correspond to the main body of the text. For example, the "Sample Data from Progressively Stable Extreme-Cases (in original units)" refers to the data of progressively stable extreme-cases in Iraq. Similarly, when a network's parameters are given, the title of the section indicates how its information relates to the text. For instance, "Parameters for Network Trained on Actual Data" refers to the parameters of the network trained on actual data of the Iraq context. Inference data is similarly labeled in accordance with its appearance in the main body of the text.

C.1. Example Macro Used for Progressively Unstable Extreme-Cases

```
''''''''''''''''''Begin Get Most Recent Data Point''''''''''''''''''''''

Sheets("Data").Select

For j = 1 To 17 'this is for 16 metrics starting in the second column
   For i = 1 To 60 'this is for 60 time steps starting in the first row
      'If Cells(62 - i, j + 1) = 0 Then
      'Cells(63, j + 1) = "true"
      If Cells(62 - i, j + 1) > 0 Then

         If Cells(62 - i, j + 1) < 1 Then
            Cells(62, j + 1) = 1 'approximate this way for ease of generating future data
         Else
            Cells(62, j + 1) = Cells(62 - i, j + 1)
         End If

      Exit For
      End If
   Next i
Next j

''''''''''''''''''End Get Most Recent Data Point''''''''''''''''''''''
```

```
"""""""""""""""""Begin Get/Enact Function's Effects on Battlefield"""""""""""""""""""""

Sheets("Effects_on_Battlefield").Select


Chance_of_Action_Effecting_Component = 0.3 'An assumed rate of efficacy for each
function.
Chance_of_Detrimental_Noise_Effecting_Component = 0.6 'Also assumed

For k = 1 To 60
   For j = 1 To 17 'Not to 16 because data starts in second column
      For i = 1 To 9

         If Cells(i + 2, j + 1) = "-" Then

            Random = Rnd 'Get a random number

            If Chance_of_Action_Effecting_Component > Random Then
               Sheets("Data").Select
               Cells(62 + k, j + 1) = Cells(61 + k, j + 1) - 0.05 * Cells(61 + k, j + 1)
'Assume a reduction by 5% of the given component of the battlefield

               If Chance_of_Detrimental_Noise_Effecting_Component > Random Then
               Cells(62 + k, j + 1) = Cells(62 + k, j + 1) + 0.2 * Cells(62 + k, j + 1) 'Assume
that noise will add 20% of component's value
               End If
```

```
            Sheets("Effects_on_Battlefield").Select
        Else
            Sheets("Data").Select
            Cells(62 + k, j + 1) = Cells(61 + k, j + 1)

            If Chance_of_Detrimental_Noise_Effecting_Component > Random Then
            Cells(62 + k, j + 1) = Cells(62 + k, j + 1) + 0.2 * Cells(62 + k, j + 1) 'Assume
that noise will add 20% of component's value
            End If

            Sheets("Effects_on_Battlefield").Select
        End If

    End If

    If Cells(i + 2, j + 1) = "+" Then

        Random = Rnd 'Get a random number

        If Chance_of_Action_Effecting_Component > Random Then
            Sheets("Data").Select
            Cells(62 + k, j + 1) = Cells(61 + k, j + 1) + 0.05 * Cells(61 + k, j + 1)
'Assume an increase by 5% of the given component of the battlefield

            If Chance_of_Detrimental_Noise_Effecting_Component > Random Then
            Cells(62 + k, j + 1) = Cells(62 + k, j + 1) - 0.2 * Cells(62 + k, j + 1) 'Assume
that noise will subtract 20% of component's value
```

```
                End If

                Sheets("Effects_on_Battlefield").Select
            Else
                Sheets("Data").Select
                Cells(62 + k, j + 1) = Cells(61 + k, j + 1)

                If Chance_of_Detrimental_Noise_Effecting_Component > Random Then
                Cells(62 + k, j + 1) = Cells(62 + k, j + 1) - 0.2 * Cells(62 + k, j + 1) 'Assume
that noise will subtract 20% of component's value
                End If

                Sheets("Effects_on_Battlefield").Select
            End If

        End If

    Next i
  Next j
Next k

"""""""""""""""""End Get/Enact Function's Effects on Battlefield"""""""""""""""""""""
```

C.2. Sample Data from Progressively Stable Extreme-Cases

(in original units)

	Metric #															
time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	750	8	20	11	1	3	2	1	1	20	1	164852	2.24	1.87	4180	28
1	855	8	19	10	1	3	2	1	1	19	1	173095	2.35	1.87	4180	28
2	812	7	18	10	1	3	2	1	1	18	1	181749	2.35	1.96	4180	27
3	772	7	17	9	1	3	2	1	1	17	1	190837	2.47	2.06	3511	25
4	733	7	17	9	1	3	2	1	1	17	1	190837	2.47	2.06	3687	24
5	696	6	17	9	1	3	2	1	1	17	1	200379	2.07	2.06	3871	24
6	662	6	16	8	1	3	2	1	1	16	1	200379	2.18	2.06	4065	23
7	629	6	15	8	1	3	2	1	1	15	1	200379	2.29	2.16	4268	23
8	597	6	15	7	1	3	2	1	1	15	1	210398	2.40	1.82	4268	22
9	597	5	15	8	1	2	2	1	1	15	1	220917	2.52	1.91	4268	21
10	681	5	14	8	1	2	2	1	1	14	1	231963	2.12	2.00	4481	20
11	647	5	14	8	1	2	2	1	1	14	1	243561	2.22	2.11	4705	19
12	737	5	13	7	1	2	2	1	1	13	1	204592	2.34	2.21	4941	19
13	737	6	13	7	1	2	2	1	1	13	1	204592	1.96	1.96	5188	18
14	737	6	13	7	1	2	2	1	1	13	1	214821	2.06	2.06	5188	17
15	700	6	12	7	1	2	2	1	1	12	1	180450	2.16	2.16	5447	16
16	700	5	12	6	1	2	2	1	1	12	1	180450	2.27	2.27	5719	15
17	798	5	11	6	1	2	2	1	1	11	1	189472	2.38	2.38	5719	17
18	758	5	11	6	1	2	2	1	1	11	1	198946	2.50	2.50	6005	16
19	865	5	11	6	1	2	2	1	1	11	1	198946	2.63	2.63	6306	16
20	821	4	11	6	1	2	2	1	1	11	1	208893	2.63	2.49	6621	15
21	780	4	10	5	1	2	2	1	1	10	1	208893	2.63	2.61	6952	14
22	780	4	10	5	0	2	2	1	1	10	1	219338	2.63	2.63	7300	14
23	741	5	10	6	0	1	2	0	1	10	1	219338	2.76	2.76	7300	16
24	741	4	10	5	0	1	2	0	1	10	1	184244	2.90	2.90	7665	15
25	704	4	10	5	0	1	2	0	1	10	1	193456	3.04	3.04	6438	15
26	669	4	10	5	0	1	1	0	1	10	1	203129	3.04	3.04	6760	15
27	669	4	9	5	0	1	1	0	1	9	1	203129	3.20	3.20	7098	14
28	763	4	9	5	0	1	1	0	1	9	1	213285	3.35	3.35	7098	14
29	725	3	9	5	0	1	1	0	1	9	1	179160	3.52	3.52	7453	14
30	688	3	9	5	0	1	1	0	1	9	1	179160	3.52	3.52	7826	13
31	654	3	8	5	0	1	1	0	1	8	1	188118	3.70	3.70	8217	13
32	621	3	8	5	0	1	1	0	1	8	1	197523	3.70	3.70	8628	12
33	708	3	8	4	0	1	1	0	1	8	0	207400	3.88	3.88	9059	12
34	708	3	8	4	0	1	1	0	1	8	0	217770	4.08	4.08	7610	12
35	673	3	7	4	0	1	1	0	1	7	0	228658	4.28	4.28	7990	12
36	639	3	7	4	0	1	1	0	1	7	0	240091	4.50	4.50	8390	11
37	639	3	7	4	0	1	1	0	1	7	0	252096	3.78	3.78	8809	11
38	729	3	7	3	0	1	1	0	1	7	0	211760	3.97	3.97	9250	11
39	692	3	7	3	0	1	1	0	1	7	0	211760	4.16	4.16	9712	10
40	658	3	7	3	0	1	1	0	1	7	0	222348	4.37	4.37	8158	10
41	750	2	7	3	0	1	1	0	1	7	0	186773	4.37	4.37	8566	10
42	712	2	6	3	0	1	1	0	1	6	1	156889	4.37	4.37	7196	11
43	677	2	6	3	0	1	1	0	1	6	1	164733	3.67	3.67	7555	10
44	643	2	6	3	0	1	1	0	0	6	0	172970	3.86	3.86	7933	10
45	643	2	5	3	0	1	1	0	0	5	0	181619	3.24	3.24	7933	10
46	611	2	6	2	0	1	1	0	0	6	0	190700	2.72	2.72	8330	10
47	580	2	6	2	0	1	1	0	0	6	0	200234	2.29	2.29	8746	9
48	580	2	6	2	0	1	1	0	0	6	0	210246	2.29	2.29	9184	9
49	551	2	5	2	0	0	1	0	0	5	0	220759	2.40	2.40	7714	8
50	523	2	5	2	0	0	1	0	0	5	0	185437	2.52	2.52	8100	8
51	497	2	5	2	0	1	1	0	0	5	0	194709	2.52	2.52	8505	8
52	472	2	5	2	0	1	1	0	0	5	0	194709	2.65	2.65	7144	7
53	472	1	5	2	0	1	1	0	0	5	0	204444	2.65	2.65	7501	7
54	472	1	4	2	0	1	1	0	0	4	0	214667	2.78	2.78	7876	7
55	539	1	4	2	0	1	1	0	0	4	0	214667	2.78	2.78	8270	6
56	539	1	4	2	0	1	1	0	0	4	0	214667	2.92	2.92	8270	6
57	512	1	4	2	0	0	0	0	0	4	0	225400	3.06	3.06	8270	6
58	486	1	4	2	0	0	0	0	0	4	0	225400	2.57	2.57	8270	6
59	462	1	4	2	0	1	0	0	0	4	0	225400	2.16	2.16	8684	6
60	439	1	3	2	0	1	0	0	0	3	0	225400	1.82	1.82	9118	5

C.3. *N = 16* Infinity-Normalized $V_{properties}$ on Iraq Context (normalized units)

	Metric #							
time step	1	2	3	4	5	6	7	8
0	0.23348611	0.00014493	0.27007299	0.00012195	0.00052632	0.00083333	0.00071429	0.17948718
1	0.27662443	0.00014493	0.21897810	0.00012195	0.00052632	0.00083333	0.28571429	0.00025641
2	0.25208951	0.01449275	0.34306569	0.04878049	0.00052632	0.00083333	0.64285714	0.00025641
3	0.34834187	0.05797101	0.26277372	0.08536585	0.00052632	0.00083333	0.14285714	0.00025641
4	0.23186843	0.04347826	0.22627737	0.07317073	0.00052632	0.16666667	0.14285714	0.02564103
5	0.22243192	0.18840580	0.32116788	0.15853659	0.00052632	0.33333333	0.14285714	0.00025641
6	0.18252898	0.08695652	0.59854015	0.24390244	0.00052632	0.08333333	0.07142857	1.00000000
7	0.22027501	0.20289855	0.29197080	0.21951220	0.05263158	0.16666667	0.00071429	0.00025641
8	0.22404961	0.13043478	0.33576642	0.24390244	0.15789474	0.33333333	0.07142857	0.35897436
9	0.25289836	0.24637681	0.15328467	0.10975610	0.00052632	0.16666667	0.00071429	0.05128205
10	0.32084120	0.13043478	0.36496350	0.23170732	0.00052632	0.33333333	0.00071429	0.00025641
11	0.54300350	0.13043478	0.98540146	0.26829268	0.52631579	0.58333333	1.00000000	0.05128205
12	0.43866271	0.13043478	0.58394161	0.25609756	0.10526316	1.00000000	0.14285714	0.00025641
13	0.27527635	0.27536232	0.30656934	0.14634146	0.10526316	0.58333333	0.07142857	0.00025641
14	0.25128067	0.15942029	0.39416058	0.20731707	0.10526316	0.58333333	0.14285714	0.00025641
15	0.40900512	0.18840580	0.47445255	0.19512195	0.00052632	0.16666667	0.28571429	0.05128205
16	0.38662712	0.23188406	0.58394161	0.18292683	0.57894737	0.33333333	0.14285714	0.00025641
17	0.35831761	0.24637681	0.46715328	0.14634146	1.00000000	0.16666667	0.28571429	0.05128205
18	0.71124292	0.15942029	1.00000000	0.21951220	0.31578947	0.33333333	0.28571429	0.00025641
19	0.35939606	0.24637681	0.52554745	0.17073171	0.10526316	0.08333333	0.00071429	0.05128205
20	0.39040173	0.40579710	0.77372263	0.35365854	0.15789474	0.25000000	0.57142857	0.84615385
21	0.43111351	0.26086957	0.42335766	0.30487805	0.05263158	0.08333333	0.00071429	0.00025641
22	0.35939606	0.18840580	0.25547445	0.15853659	0.36842105	0.08333333	0.00071429	0.00025641
23	0.32353734	0.30434783	0.37956204	0.24390244	0.36842105	0.41666667	0.14285714	0.00025641
24	0.47910488	0.52173913	0.57664234	0.40243902	0.52631579	0.50000000	0.14285714	0.05128205
25	0.40900512	0.49275362	0.56934307	0.43902439	0.42105263	0.16666667	0.21428571	0.05128205
26	0.44702076	0.37681159	0.39416058	0.43902439	0.10526316	0.25000000	0.00071429	0.00025641
27	0.89053653	0.39130435	0.62043796	0.48780488	0.36842105	0.08333333	0.00071429	0.00025641
28	0.52952278	0.66666667	0.35766423	0.45121951	0.00052632	0.16666667	0.00071429	0.00025641
29	0.37098949	0.56521739	0.70072993	0.69512195	0.10526316	0.58333333	0.00071429	0.00025641
30	0.44216770	0.59420290	0.61313869	0.48780488	0.31578947	0.00083333	0.00071429	0.05128205
31	0.36344028	0.30434783	0.49635036	0.51219512	0.15789474	0.16666667	0.07142857	0.05128205
32	0.47937449	0.43478261	0.44525547	0.29268293	0.15789474	0.00083333	0.07142857	0.33333333
33	0.58371529	0.56521739	0.39416058	0.43902439	0.10526316	0.08333333	0.00071429	0.00025641
34	0.64114317	0.53623188	0.22627737	0.14634146	0.05263158	0.25000000	0.07142857	0.00025641
35	0.61579941	0.57971014	0.55474453	0.54878049	0.05263158	0.08333333	0.07142857	0.05128205
36	0.71960097	0.81159420	0.50364964	0.43902439	0.10526316	0.00083333	0.00071429	0.10256410
37	0.84901591	0.82608696	0.44525547	0.40243902	0.00052632	0.08333333	0.00071429	0.00025641
38	0.96791588	0.76811594	0.31386861	0.25609756	0.15789474	0.00083333	0.07142857	0.00025641
39	0.81126988	0.75362319	0.47445255	0.35365854	0.00052632	0.00083333	0.00071429	0.05128205
40	0.90186034	0.82608696	0.52554745	0.35365854	0.21052632	0.08333333	0.07142857	0.00025641
41	1.00000000	0.81159420	0.77372263	0.63414634	0.00052632	0.00083333	0.07142857	0.00025641
42	0.93340523	0.94202899	0.50364964	0.46341463	0.00052632	0.00083333	0.00071429	0.05128205
43	0.78565651	1.00000000	0.82481752	0.82926829	0.00052632	0.08333333	0.07142857	0.12820513
44	0.94365058	0.63768116	0.60583942	0.42682927	0.00052632	0.16666667	0.00071429	0.35897436
45	0.72795902	0.81159420	0.59124088	0.30487805	0.10526316	0.16666667	0.00071429	0.23076923
46	0.64707468	0.73913043	0.59124088	0.60975610	0.00052632	0.16666667	0.00071429	0.00025641
47	0.67403613	0.76811594	0.75912409	0.73170732	0.00052632	0.08333333	0.07142857	0.00025641
48	0.70099757	0.60869565	0.91970803	1.00000000	0.00052632	0.00083333	0.00071429	0.05128205
49	0.52574818	0.56521739	0.73722628	0.70731707	0.00052632	0.00083333	0.28571429	0.00025641
50	0.63359396	0.62318841	0.56934307	0.53658537	0.00052632	0.16666667	0.14285714	0.02564103
51	0.53922890	0.37681159	0.61313869	0.39024390	0.00052632	0.00083333	0.28571429	0.48717949
52	0.29657590	0.43478261	0.47445255	0.30487805	0.05263158	0.00083333	0.21428571	0.00025641
53	0.25613373	0.49275362	0.27737226	0.24390244	0.00052632	0.16666667	0.00071429	0.00025641
54	0.20221084	0.31884058	0.26277372	0.31707317	0.00052632	0.00083333	0.00071429	0.00025641
55	0.20221084	0.33333333	0.16788321	0.10975610	0.00052632	0.00083333	0.00071429	0.00025641
56	0.16176867	0.34782609	0.29197080	0.28048780	0.00052632	0.00083333	0.07142857	0.00025641
57	0.18873012	0.30434783	0.21167883	0.20731707	0.00052632	0.00083333	0.07142857	0.00025641
58	0.20221084	0.40579710	0.28467153	0.31707317	0.00052632	0.25000000	0.07142857	0.02564103
59	0.00000000	0.11594203	0.19708029	0.13414634	0.00052632	0.25000000	0.14285714	0.00025641

	Metric #							
time step	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
0	0.06451613	0.27007299	0.00000000	0.94535519	0.11933174	0.00518135	0.10288066	0.00000000
1	0.15053763	0.21897810	0.20000000	0.93442623	0.26849642	0.10362694	0.65699588	1.00000000
2	0.16129032	0.35036496	0.06666667	0.92896175	0.36793954	0.16683938	0.66584362	0.00000000
3	0.07526882	0.25547445	0.16666667	0.87978142	0.57478123	0.33471503	0.67139918	0.98333333
4	0.08602151	0.22627737	0.06666667	0.85245902	0.68516309	0.50932642	0.72901235	0.00000000
5	0.15053763	0.32116788	0.13333333	0.85245902	0.81742243	0.59533679	0.81234568	0.76666667
6	0.08602151	0.59854015	0.30000000	0.80273224	0.83532220	0.78963731	0.73703704	0.00000000
7	0.04301075	0.29197080	0.30000000	0.80054645	0.91487669	0.79844560	0.70514403	0.81666667
8	0.04301075	0.34306569	0.06666667	0.80655738	0.97056484	0.79637306	0.77325103	0.51666667
9	0.03225806	0.14598540	0.06666667	0.75956284	0.90533015	0.71606218	0.84876543	0.63333333
10	0.12903226	0.37956204	0.20000000	0.84153005	0.96857597	0.94559585	0.83127572	0.71666667
11	0.77419355	0.99270073	0.13333333	0.88524590	0.94828958	0.93471503	0.78662551	0.51666667
12	0.26881720	0.58394161	0.23333333	0.88524590	0.75059666	0.71502591	0.80288066	0.56666667
13	0.16129032	0.30656934	0.40000000	0.87978142	0.91288783	0.59481865	0.88333333	0.53333333
14	0.17204301	0.39416058	0.56666667	0.88524590	0.87509944	0.72849741	0.94320988	0.60000000
15	0.35483871	0.48175182	0.70000000	0.89453552	0.84009547	0.57720207	0.96851852	0.58333333
16	0.39784946	0.58394161	0.66666667	0.88852459	1.00000000	0.88238342	0.91913580	0.58333333
17	0.20430108	0.46715328	0.36666667	0.88524590	0.97852029	0.79896373	0.83827160	0.58333333
18	1.00000000	1.00000000	1.00000000	0.88524590	0.77565632	0.68393782	0.65823045	0.58333333
19	0.44086022	0.52554745	0.56666667	0.94535519	0.85918854	0.78756477	0.69547325	0.53333333
20	0.11827957	0.78102190	0.43333333	0.95792350	0.83532220	0.70829016	0.67674897	0.50000000
21	0.16129032	0.42335766	0.43333333	0.98360656	0.83532220	0.74145078	0.74300412	0.55000000
22	0.10752688	0.25547445	0.33333333	0.93989071	0.83134447	0.72227979	0.74629630	0.51666667
23	0.12903226	0.37956204	0.16666667	0.89617486	0.85123309	0.72435233	0.69753086	0.58333333
24	0.15053763	0.58394161	0.33333333	0.87978142	0.83532220	0.67772021	0.76378601	0.63333333
25	0.19354839	0.56934307	0.33333333	0.86338798	0.86316627	0.71347150	0.85452675	0.58333333
26	0.04301075	0.39416058	0.30000000	0.87978142	0.86316627	0.80310881	0.91481481	0.55000000
27	0.29032258	0.62043796	0.30000000	0.87978142	0.85918854	0.77927461	0.83312757	0.51666667
28	0.03225806	0.35766423	0.30000000	0.87431694	0.83929992	0.82901554	0.85576132	0.46666667
29	0.11827957	0.70072993	0.30000000	0.95081967	0.75974543	0.64196891	0.75823045	0.46666667
30	0.25806452	0.61313869	0.00000000	1.00000000	0.78758950	0.60518135	0.76995885	0.58333333
31	0.09677419	0.49635036	0.10000000	1.00000000	0.76372315	0.55492228	0.78189300	0.63333333
32	0.10752688	0.45255474	0.33333333	0.85792350	0.68814638	0.54404145	0.74897119	0.46666667
33	0.07526882	0.40145985	0.30000000	0.83606557	0.72792363	0.76165803	0.76131687	0.56666667
34	0.09677419	0.22627737	0.50000000	0.83606557	0.83532220	0.68393782	0.82304527	0.55000000
35	0.16129032	0.55474453	0.20000000	0.83060109	0.85123309	0.82901554	0.76131687	0.65000000
36	0.18279570	0.50364964	0.43333333	0.83060109	0.84725537	0.78238342	0.80246914	0.58333333
37	0.24731183	0.44525547	0.26666667	0.80273224	0.91487669	0.86528497	0.90534979	0.60000000
38	0.13978495	0.31386861	0.23333333	0.81420765	0.88305489	0.87046632	0.90534979	0.60000000
39	0.31182796	0.47445255	0.06666667	0.85792350	0.89101034	0.87046632	0.91152263	0.46666667
40	0.27956989	0.52554745	0.33333333	0.88524590	0.93078759	0.85492228	0.82304527	0.46666667
41	0.49462366	0.77372263	0.13333333	0.88087432	0.89896579	0.80310881	0.82304527	0.41666667
42	0.23655914	0.51094891	0.36666667	0.86338798	0.83532220	0.74611399	0.76131687	0.55000000
43	0.26881720	0.81751825	0.16666667	0.84808743	0.85521082	0.75129534	0.72016461	0.55000000
44	0.29032258	0.60583942	0.16666667	0.80136612	0.66030231	0.67357513	0.73868313	0.43333333
45	0.35483871	0.59124088	0.26666667	0.81426230	0.82736675	0.77720207	0.74074074	0.63333333
46	0.20430108	0.59124088	0.10000000	0.84811475	0.82736675	0.81865285	0.74074074	0.41666667
47	0.36559140	0.75912409	0.63333333	0.86992350	0.85123309	0.77720207	0.78806584	0.48333333
48	0.38709677	0.91970803	0.46666667	0.88421858	0.80747812	0.84974093	0.76543210	0.45000000
49	0.33333333	0.73722628	0.46666667	0.92089617	0.79554495	0.76165803	0.86419753	0.66666667
50	0.19354839	0.56934307	0.40000000	0.93720219	0.82338902	0.88601036	0.86831276	0.55000000
51	0.21505376	0.61313869	0.00033333	0.94909836	0.75974543	0.87564767	0.90123457	0.66666667
52	0.13978495	0.47445255	0.03333333	0.98513115	0.91487669	0.98445596	1.00000000	0.48333333
53	0.07526882	0.27737226	0.03333333	0.99818579	0.93078759	0.98963731	0.97222222	0.50000000
54	0.06451613	0.27007299	0.03333333	0.94857377	0.94669849	0.97409326	0.85185185	0.61666667
55	0.05376344	0.16788321	0.03333333	0.93421311	0.96260939	1.00000000	0.87860082	0.46666667
56	0.11827957	0.29197080	0.03333333	0.91586885	0.89101034	1.00000000	0.82921811	0.53333333
57	0.07526882	0.21167883	0.06666667	0.91199454	0.95067621	1.00000000	0.81275720	0.43333333
58	0.05376344	0.27737226	0.03333333	0.90147541	0.94669849	1.00000000	0.86831276	0.43333333
59	0.01075269	0.14598540	0.00000000	0.90083060	0.89101034	0.96891192	0.86008230	0.56666667

C.4. Parameters for Network Trained on Actual Data

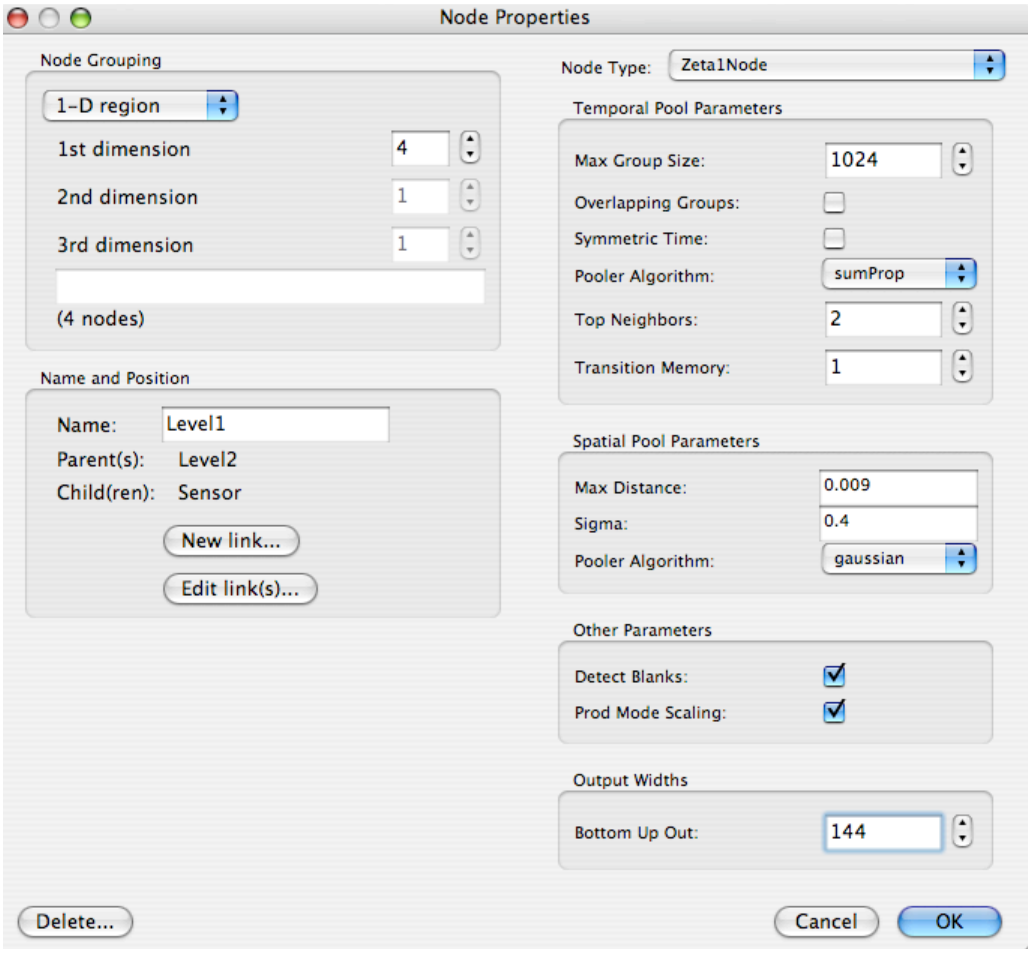


Figure 111: Bottom-Level Node Settings

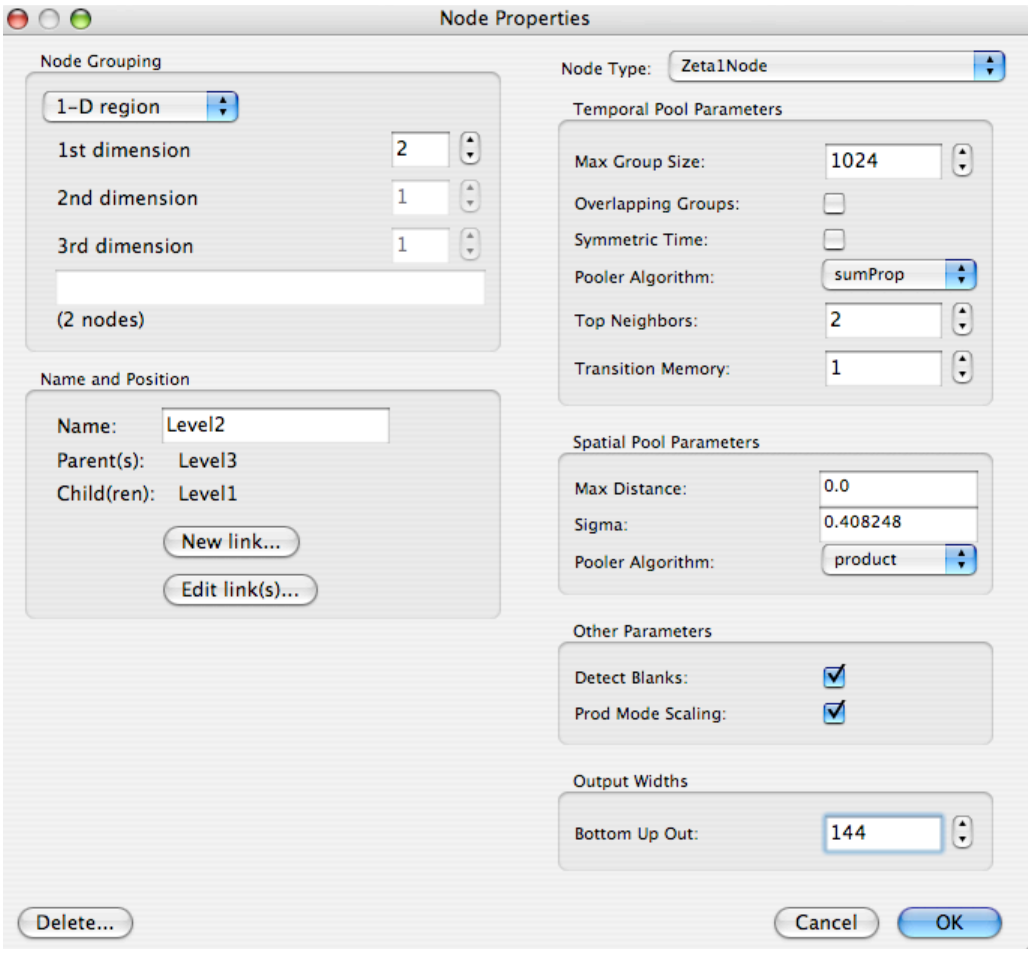


Figure 112: Second-Level Node Settings

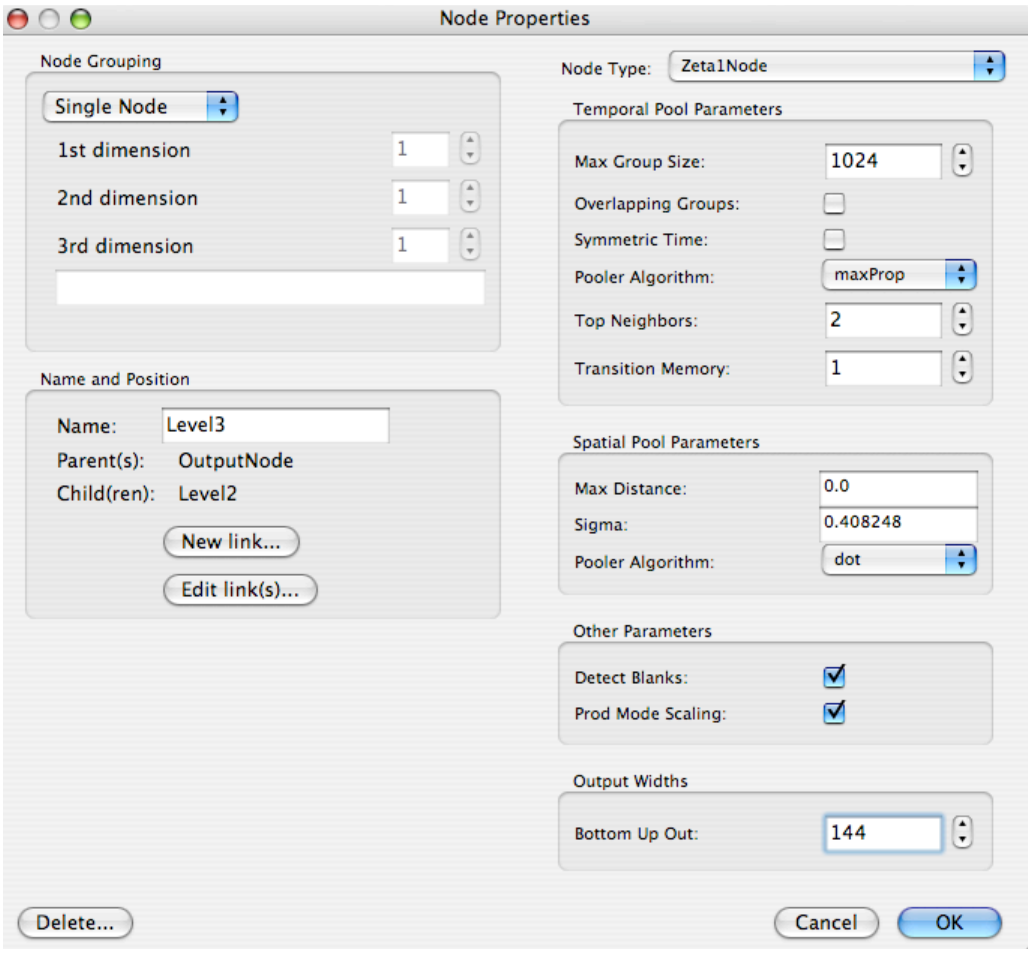


Figure 113: Top-Level Node Settings

C.5. Inference History on Progressively Stable Data

(Trained on Actual Data)

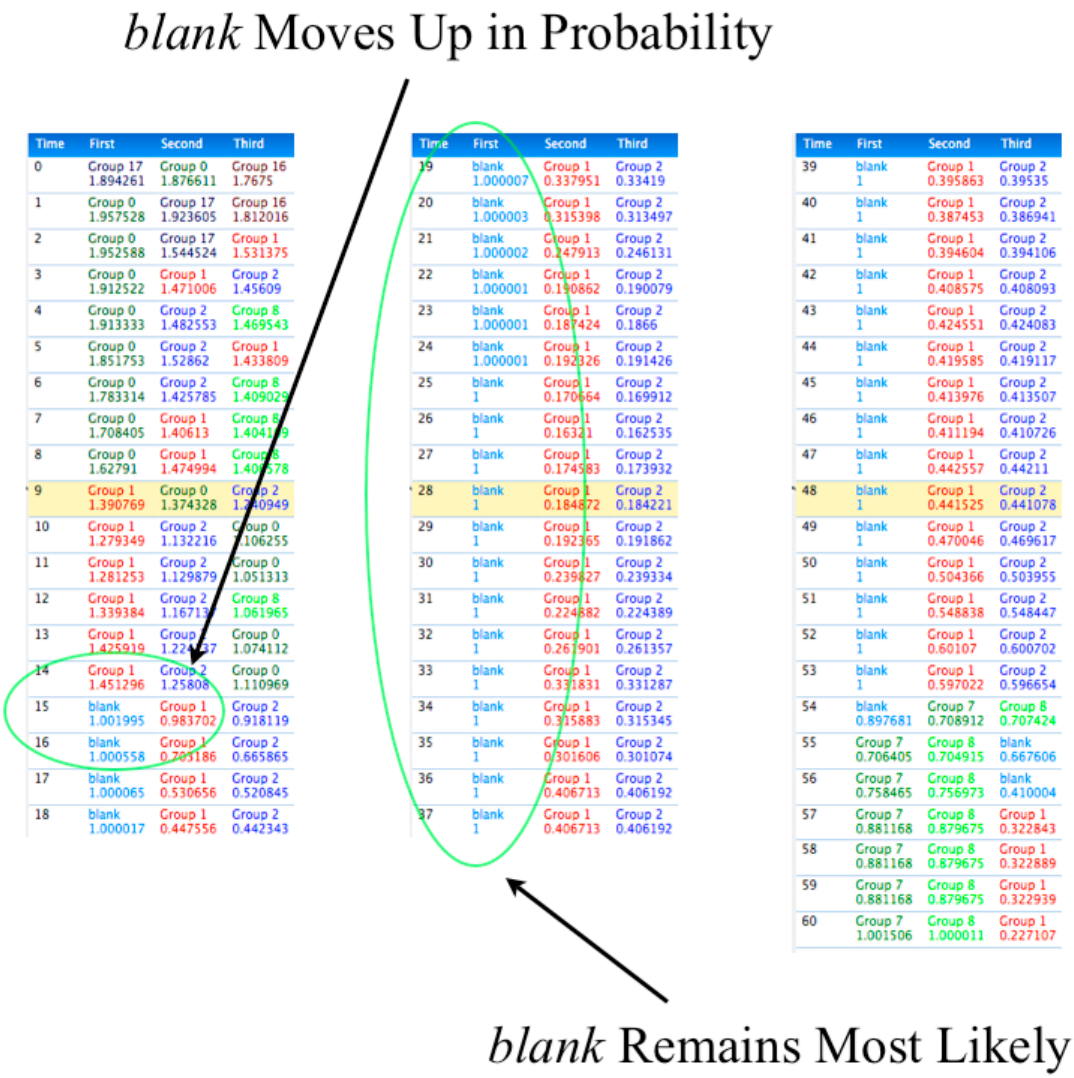
Time	First	Second	Third
0	Group 0 1.825213	Group 2 1.638828	Group 1 1.569814
1	Group 0 1.808792	Group 2 1.617977	Group 1 1.548099
2	Group 0 1.829446	Group 2 1.597264	Group 1 1.540601
3	Group 0 1.846811	Group 2 1.568537	Group 1 1.525939
4	Group 0 1.843979	Group 2 1.550884	Group 1 1.510503
5	Group 0 1.814129	Group 2 1.51425	Group 1 1.482992
6	Group 0 1.81566	Group 2 1.498174	Group 1 1.466441
7	Group 0 1.820805	Group 2 1.483248	Group 1 1.459091
8	Group 0 1.823317	Group 2 1.483338	Group 1 1.424571
9	Group 0 1.830257	Group 2 1.470916	Group 1 1.421794
10	Group 0 1.769923	Group 2 1.43649	Group 1 1.408011
11	Group 0 1.770953	Group 2 1.403455	Group 1 1.384048
12	Group 0 1.752524	Group 2 1.435533	Group 1 1.421932
13	Group 0 1.727085	Group 2 1.469017	Group 1 1.439956
14	Group 0 1.720371	Group 2 1.440166	Group 1 1.42083
15	Group 0 1.735883	Group 2 1.452974	Group 1 1.441291
16	Group 0 1.735022	Group 2 1.428558	Group 1 1.425794
17	Group 0 1.707347	Group 1 1.397526	Group 2 1.395833
18	Group 0 1.710728	Group 1 1.379951	Group 2 1.369192

Time	First	Second	Third
19	Group 0 1.670888	Group 1 1.365173	Group 2 1.345286
20	Group 0 1.682104	Group 1 1.346436	Group 2 1.338902
21	Group 0 1.688742	Group 1 1.343765	Group 2 1.325315
22	Group 0 1.681441	Group 1 1.323436	Group 2 1.305162
23	Group 0 1.722244	Group 1 1.35301	Group 2 1.329225
24	Group 0 1.726736	Group 1 1.374167	Group 2 1.339241
25	Group 0 1.728823	Group 1 1.357745	Group 17 1.323325
26	Group 0 1.74034	Group 17 1.345072	Group 1 1.339271
27	Group 0 1.741043	Group 17 1.401525	Group 1 1.328346
28	Group 0 1.706228	Group 17 1.425871	Group 16 1.335567
29	Group 0 1.740949	Group 17 1.536485	Group 16 1.431267
30	Group 0 1.744621	Group 17 1.550307	Group 16 1.435494
31	Group 0 1.745434	Group 17 1.663155	Group 16 1.531694
32	Group 0 1.731123	Group 17 1.665054	Group 16 1.532521
33	Group 17 1.67065	Group 0 1.60421	Group 16 1.565964
34	Group 17 1.66078	Group 0 1.584077	Group 16 1.551002
35	Group 17 1.642451	Group 0 1.554963	Group 16 1.523767
36	Group 17 1.619615	Group 0 1.522295	Group 16 1.498998
37	Group 0 1.637792	Group 17 1.623836	Group 16 1.507728

Time	First	Second	Third
38	Group 17 1.611296	Group 0 1.533685	Group 16 1.516289
39	Group 17 1.596241	Group 0 1.508674	Group 16 1.497503
40	Group 17 1.586926	Group 0 1.491266	Group 16 1.477161
41	Group 17 1.574537	Group 0 1.499431	Group 16 1.484733
42	Group 17 1.585176	Group 0 1.501331	Group 16 1.466938
43	Group 0 1.62986	Group 17 1.481538	Group 16 1.361333
44	Group 0 1.592768	Group 17 1.567585	Group 16 1.435993
45	Group 0 1.590843	Group 17 1.273439	Group 1 1.169812
46	Group 0 1.572105	Group 1 1.15704	Group 2 1.12943
47	Group 0 1.594107	Group 2 1.137196	Group 1 1.136052
48	Group 0 1.590103	Group 2 1.120361	Group 1 1.11869
49	Group 0 1.575128	Group 1 1.093664	Group 2 1.083248
50	Group 0 1.571741	Group 1 1.114559	Group 2 1.096121
51	Group 0 1.568115	Group 1 1.098025	Group 2 1.080548
52	Group 0 1.555113	Group 1 1.08955	Group 2 1.059833
53	Group 0 1.551475	Group 1 1.077077	Group 2 1.048351
54	Group 0 1.536349	Group 17 1.074337	Group 1 1.061015
55	Group 0 1.535691	Group 17 1.075603	Group 1 1.067064
56	Group 0 1.523979	Group 17 1.107739	Group 1 1.062511
57	Group 0 1.51337	Group 17 1.145356	Group 1 1.043325
58	Group 0 1.548882	Group 1 1.032301	Group 2 1.009429
59	Group 0 1.587707	Group 2 1.029794	Group 1 1.027166
60	Group 0 1.623911	Group 2 1.04117	Group 1 1.016223

C.6. Inference History on Progressively Stable Data

(Trained on Actual Data)



Time	First	Second	Third
0	Group 17 1.894261	Group 0 1.876611	Group 16 1.7675
1	Group 0 1.957528	Group 17 1.923605	Group 16 1.812016
2	Group 0 1.952588	Group 17 1.544524	Group 1 1.531375
3	Group 0 1.912522	Group 1 1.471006	Group 2 1.45609
4	Group 0 1.913333	Group 2 1.482553	Group 8 1.469543
5	Group 0 1.851753	Group 2 1.52862	Group 1 1.433809
6	Group 0 1.783314	Group 2 1.425785	Group 8 1.409029
7	Group 0 1.708405	Group 1 1.40613	Group 8 1.404179
8	Group 0 1.62791	Group 1 1.474994	Group 8 1.405578
9	Group 1 1.390769	Group 0 1.374328	Group 2 1.340949
10	Group 1 1.279349	Group 2 1.132216	Group 0 1.106255
11	Group 1 1.281253	Group 2 1.129879	Group 0 1.051313
12	Group 1 1.339384	Group 2 1.167137	Group 8 1.061965
13	Group 1 1.425919	Group 2 1.224137	Group 0 1.074112
14	Group 1 1.451296	Group 2 1.25808	Group 0 1.110969
15	blank 1.001995	Group 1 0.983702	Group 2 0.918119
16	blank 1.000558	Group 1 0.793186	Group 2 0.665865
17	blank 1.000065	Group 1 0.530656	Group 2 0.520845
18	blank 1.000017	Group 1 0.447556	Group 2 0.442343
19	blank 1.000007	Group 1 0.337951	Group 2 0.33419
20	blank 1.000003	Group 1 0.315398	Group 2 0.313497
21	blank 1.000002	Group 1 0.247913	Group 2 0.246131
22	blank 1.000001	Group 1 0.190862	Group 2 0.190079
23	blank 1.000001	Group 1 0.187424	Group 2 0.1866
24	blank 1.000001	Group 1 0.192326	Group 2 0.191426
25	blank 1	Group 1 0.170664	Group 2 0.169912
26	blank 1	Group 1 0.16321	Group 2 0.162535
27	blank 1	Group 1 0.174583	Group 2 0.173932
28	blank 1	Group 1 0.184872	Group 2 0.184221
29	blank 1	Group 1 0.192365	Group 2 0.191862
30	blank 1	Group 1 0.239827	Group 2 0.239334
31	blank 1	Group 1 0.224882	Group 2 0.224389
32	blank 1	Group 1 0.261901	Group 2 0.261357
33	blank 1	Group 1 0.331831	Group 2 0.331287
34	blank 1	Group 1 0.315883	Group 2 0.315345
35	blank 1	Group 1 0.301606	Group 2 0.301074
36	blank 1	Group 1 0.406713	Group 2 0.406192
37	blank 1	Group 1 0.406713	Group 2 0.406192
39	blank 1	Group 1 0.395863	Group 2 0.39535
40	blank 1	Group 1 0.387453	Group 2 0.386941
41	blank 1	Group 1 0.394604	Group 2 0.394106
42	blank 1	Group 1 0.408575	Group 2 0.408093
43	blank 1	Group 1 0.424551	Group 2 0.424083
44	blank 1	Group 1 0.419585	Group 2 0.419117
45	blank 1	Group 1 0.413976	Group 2 0.413507
46	blank 1	Group 1 0.411194	Group 2 0.410726
47	blank 1	Group 1 0.442557	Group 2 0.44211
48	blank 1	Group 1 0.441525	Group 2 0.441078
49	blank 1	Group 1 0.470046	Group 2 0.469617
50	blank 1	Group 1 0.504366	Group 2 0.503955
51	blank 1	Group 1 0.548838	Group 2 0.548447
52	blank 1	Group 1 0.60107	Group 2 0.600702
53	blank 1	Group 1 0.597022	Group 2 0.596654
54	blank 0.897681	Group 7 0.708912	Group 8 0.707424
55	Group 7 0.706405	Group 8 0.704915	blank 0.667606
56	Group 7 0.758465	Group 8 0.756973	blank 0.410004
57	Group 7 0.881168	Group 8 0.879675	Group 1 0.322843
58	Group 7 0.881168	Group 8 0.879675	Group 1 0.322889
59	Group 7 0.881168	Group 8 0.879675	Group 1 0.322939
60	Group 7 1.001506	Group 8 1.000011	Group 1 0.227107

C.7. Sample Data from Progressively Unstable Extreme-Cases

(in original units)

	Metric #							
time step	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	750	8	20	11	1	3	2	1
1	855	8	24	13	1	3	2	1
2	975	10	24	14	1	4	3	1
3	1111	10	29	16	1	4	3	2
4	1333	11	33	16	1	4	3	2
5	1520	13	39	16	1	5	3	2
6	1520	15	39	19	1	6	4	3
7	1520	18	40	22	1	7	4	3
8	1520	20	45	25	1	9	4	4
9	1520	20	53	30	1	10	5	4
10	1520	23	61	30	1	12	5	4
11	1733	27	61	30	2	12	5	4
12	2079	27	74	35	2	12	5	5
13	2079	27	74	40	2	12	6	6
14	2079	30	88	48	3	12	8	6
15	2371	35	90	57	3	12	9	6
16	2845	42	106	65	3	12	9	7
17	3243	50	116	78	3	12	11	9
18	3697	57	122	82	4	12	11	10
19	3709	68	126	82	4	12	11	13
20	3709	68	137	82	5	12	12	14
21	3709	68	137	82	6	12	12	14
22	3709	68	137	82	6	12	14	14
23	3709	68	137	82	7	12	14	16
24	3709	68	142	82	8	12	14	20
25	3709	68	143	82	9	12	14	20
26	3709	69	148	82	11	12	14	22
27	3709	69	153	82	13	12	14	25
28	3709	69	161	82	15	12	14	31
29	3709	69	163	82	17	12	14	31
30	3709	69	171	82	19	12	14	37
31	3709	69	171	82	19	12	14	37
32	3709	69	173	82	19	12	14	39
33	3709	69	173	82	19	12	14	39
34	3709	69	175	82	19	12	14	39
35	3709	69	176	82	19	12	14	39
36	3709	69	178	82	19	12	14	39
37	3709	69	178	82	19	12	14	39
38	3709	69	179	82	19	12	14	39
39	3709	69	179	82	19	12	14	39
40	3709	69	179	82	19	12	14	39
41	3709	69	182	82	19	12	14	39
42	3709	69	185	82	19	12	14	39
43	3709	69	188	82	19	12	14	39
44	3709	69	188	82	19	12	14	39
45	3709	69	188	82	19	12	14	39
46	3709	69	188	82	19	12	14	39
47	3709	69	192	82	19	12	14	39
48	3709	69	192	82	19	12	14	39
49	3709	69	196	82	19	12	14	39
50	3709	69	200	82	19	12	14	39
51	3709	69	205	82	19	12	14	39
52	3709	69	210	82	19	12	14	39
53	3709	69	210	82	19	12	14	39
54	3709	69	219	82	19	12	14	39
55	3709	69	226	82	19	12	14	39
56	3709	69	239	82	19	12	14	39
57	3709	69	249	82	19	12	14	39
58	3709	69	249	82	19	12	14	39
59	3709	69	249	82	19	12	14	39
60	3709	69	259	82	19	12	14	39

	Metric #							
time step	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
0	1	20	1	164852	2.24	1.87	4180	40
1	1	24	1	164852	2.24	1.57	4180	46
2	1	24	1	164852	1.79	1.32	4180	55
3	2	29	1	131882	1.79	1.11	3511	66
4	2	33	1	110781	1.79	1.11	3511	75
5	2	39	1	110781	1.51	0.89	3511	75
6	2	39	2	110781	1.51	0.89	2949	90
7	2	40	2	110781	1.51	0.89	2478	100
8	2	45	2	93056	1.51	0.71	2081	100
9	2	53	2	93056	1.20	0.60	2081	100
10	2	61	3	74445	0.96	0.60	2081	100
11	2	61	3	62533	0.96	0.50	1748	100
12	2	74	3	62533	0.96	0.50	1468	100
13	2	74	3	62533	0.81	0.42	1233	100
14	3	88	4	52528	0.65	0.34	1233	100
15	3	90	4	42022	0.52	0.27	1233	100
16	3	106	4	35299	0.52	0.23	1036	100
17	3	116	5	28239	0.44	0.23	1036	100
18	4	122	5	23721	0.44	0.23	870	100
19	5	126	5	18977	0.37	0.19	870	100
20	5	137	7	15940	0.37	0.19	696	100
21	5	137	7	15940	0.29	0.16	557	100
22	5	137	7	12752	0.29	0.13	446	100
23	5	137	9	12752	0.29	0.11	356	100
24	6	142	11	12752	0.23	0.11	299	100
25	6	143	11	10712	0.23	0.09	252	100
26	6	148	11	8570	0.23	0.07	252	100
27	6	153	13	8570	0.20	0.06	201	100
28	7	161	13	7198	0.20	0.06	201	100
29	7	163	15	6047	0.16	0.05	161	100
30	7	171	19	6047	0.13	0.04	135	100
31	7	171	19	5079	0.11	0.04	135	100
32	7	173	21	4267	0.09	0.04	135	100
33	7	173	25	4267	0.08	0.03	114	100
34	9	175	25	3413	0.07	0.03	91	100
35	10	176	25	2731	0.05	0.03	76	100
36	12	178	30	2294	0.04	0.02	61	100
37	12	178	30	2294	0.04	0.02	61	100
38	13	179	30	1835	0.04	0.02	49	100
39	13	179	30	1541	0.04	0.02	49	100
40	13	179	30	1541	0.03	0.02	39	100
41	16	182	30	1295	0.02	0.01	31	100
42	19	185	30	1088	0.02	0.01	26	100
43	22	188	30	914	0.02	0.01	21	100
44	22	188	30	914	0.02	0.01	17	100
45	22	188	30	731	0.01	0.01	17	100
46	22	188	30	731	0.01	0.01	14	100
47	26	192	30	731	0.01	0.01	12	100
48	26	192	30	731	0.01	0.01	12	100
49	30	196	30	731	0.01	0.00	9	100
50	34	200	30	731	0.01	0.00	8	100
51	39	205	30	731	0.01	0.00	8	100
52	44	210	30	585	0.00	0.00	8	100
53	44	210	30	491	0.00	0.00	6	100
54	53	219	30	491	0.00	0.00	5	100
55	60	226	30	491	0.00	0.00	5	100
56	73	239	30	491	0.00	0.00	4	100
57	83	249	30	393	0.00	0.00	3	100
58	83	249	30	393	0.00	0.00	2	100
59	83	249	30	393	0.00	0.00	2	100

C.8. Progressively Stable-Then-Unstable Extreme-Cases (in original units)

Table 25: Progressively Stable Metrics Plus Break

	Metric #							
time step	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	750	8	20	11	1	3	2	1
1	855	8	19	10	1	3	2	1
2	812	7	18	10	1	3	2	1
3	772	7	17	9	1	3	2	1
4	733	7	17	9	1	3	2	1
5	696	6	17	9	1	3	2	1
6	662	6	16	8	1	3	2	1
7	629	6	15	8	1	3	2	1
8	597	6	15	7	1	3	2	1
9	597	5	15	8	1	2	2	1
10	681	5	14	8	1	2	2	1
11	647	5	14	8	1	2	2	1
12	737	5	13	7	1	2	2	1
13	737	6	13	7	1	2	2	1
14	737	6	13	7	1	2	2	1
15	700	6	12	7	1	2	2	1
16	700	5	12	6	1	2	2	1
17	798	5	11	6	1	2	2	1
18	758	5	11	6	1	2	2	1
19	865	5	11	6	1	2	2	1
20	821	4	11	6	1	2	2	1
21	780	4	10	5	1	2	2	1
22	780	4	10	5	0	2	2	1
23	741	5	10	6	0	1	2	0
24	741	4	10	5	0	1	2	0
25	704	4	10	5	0	1	2	0
26	669	4	10	5	0	1	1	0
27	669	4	9	5	0	1	1	0
28	763	4	9	5	0	1	1	0
29	725	3	9	5	0	1	1	0
30	688	3	9	5	0	1	1	0
31	654	3	8	5	0	1	1	0
32	621	3	8	5	0	1	1	0
33	708	3	8	4	0	1	1	0
34	708	3	8	4	0	1	1	0
35	673	3	7	4	0	1	1	0
36	639	3	7	4	0	1	1	0
37	639	3	7	4	0	1	1	0
38	729	3	7	3	0	1	1	0
39	692	3	7	3	0	1	1	0
40	658	3	7	3	0	1	1	0
41	750	2	7	3	0	1	1	0
42	712	2	6	3	0	1	1	0
43	677	2	6	3	0	1	1	0
44	643	2	6	3	0	1	1	0
45	643	2	5	3	0	1	1	0
46	611	2	6	2	0	1	1	0
47	580	2	6	2	0	1	1	0
48	580	2	6	2	0	1	1	0
49	551	2	5	2	0	0	1	0
50	523	2	5	2	0	0	1	0
51	497	2	5	2	0	1	1	0
52	472	2	5	2	0	1	1	0
53	472	1	5	2	0	1	1	0
54	472	1	4	2	0	1	1	0
55	539	1	4	2	0	1	1	0
56	539	1	4	2	0	1	1	0
57	512	1	4	2	0	0	0	0
58	486	1	4	2	0	0	0	0
59	462	1	4	2	0	1	0	0
60	439	1	3	2	0	1	0	0
61	0	0	0	0	0	0	0	0

Metric #							
9	**10**	**11**	**12**	**13**	**14**	**15**	**16**
1	20	1	164852	2.24	1.87	4180	28
1	19	1	173095	2.35	1.87	4180	28
1	18	1	181749	2.35	1.96	4180	27
1	17	1	190837	2.47	2.06	3511	25
1	17	1	190837	2.47	2.06	3687	24
1	17	1	200379	2.07	2.06	3871	24
1	16	1	200379	2.18	2.06	4065	23
1	15	1	200379	2.29	2.16	4268	23
1	15	1	210398	2.40	1.82	4268	22
1	15	1	220917	2.52	1.91	4268	21
1	14	1	231963	2.12	2.00	4481	20
1	14	1	243561	2.22	2.11	4705	19
1	13	1	204592	2.34	2.21	4941	19
1	13	1	204592	1.96	1.96	5188	18
1	13	1	214821	2.06	2.06	5188	17
1	12	1	180450	2.16	2.16	5447	16
1	12	1	180450	2.27	2.27	5719	15
1	11	1	189472	2.38	2.38	5719	17
1	11	1	198946	2.50	2.50	6005	16
1	11	1	198946	2.63	2.63	6306	16
1	11	1	208893	2.63	2.49	6621	15
1	10	1	208893	2.63	2.61	6952	14
1	10	1	219338	2.63	2.63	7300	14
1	10	1	219338	2.76	2.76	7300	16
1	10	1	184244	2.90	2.90	7665	15
1	10	1	193456	3.04	3.04	6438	15
1	10	1	203129	3.04	3.04	6760	15
1	9	1	203129	3.20	3.20	7098	14
1	9	1	213285	3.35	3.35	7098	14
1	9	1	179160	3.52	3.52	7453	14
1	9	1	179160	3.52	3.52	7826	13
1	8	1	188118	3.70	3.70	8217	13
1	8	1	197523	3.70	3.70	8628	12
1	8	0	207400	3.88	3.88	9059	12
1	8	0	217770	4.08	4.08	7610	12
1	7	0	228658	4.28	4.28	7990	12
1	7	0	240091	4.50	4.50	8390	11
1	7	0	252096	3.78	3.78	8809	11
1	7	0	211760	3.97	3.97	9250	11
1	7	0	211760	4.16	4.16	9712	10
1	7	0	222348	4.37	4.37	8158	10
1	7	0	186773	4.37	4.37	8566	10
1	6	1	156889	4.37	4.37	7196	11
1	6	1	164733	3.67	3.67	7555	10
0	6	0	172970	3.86	3.86	7933	10
0	5	0	181619	3.24	3.24	7933	10
0	6	0	190700	2.72	2.72	8330	10
0	6	0	200234	2.29	2.29	8746	9
0	6	0	210246	2.29	2.29	9184	9
0	5	0	220759	2.40	2.40	7714	8
0	5	0	185437	2.52	2.52	8100	8
0	5	0	194709	2.52	2.52	8505	8
0	5	0	194709	2.65	2.65	7144	7
0	5	0	204444	2.65	2.65	7501	7
0	4	0	214667	2.78	2.78	7876	7
0	4	0	214667	2.78	2.78	8270	6
0	4	0	214667	2.92	2.92	8270	6
0	4	0	225400	3.06	3.06	8270	6
0	4	0	225400	2.57	2.57	8270	6
0	4	0	225400	2.16	2.16	8684	6
0	3	0	225400	1.82	1.82	9118	5
0	0	0	0	0	0	0	0

Table 26: Progressively Unstable Metrics

	Metric #							
time step	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
62	750	8	20	11	1	3	2	1
63	855	8	24	13	1	3	2	1
64	975	10	24	14	1	4	3	1
65	1111	10	29	16	1	4	3	2
66	1333	11	33	16	1	4	3	2
67	1520	13	39	16	1	5	3	2
68	1520	15	39	19	1	6	4	3
69	1520	18	40	22	1	7	4	3
70	1520	20	45	25	1	9	4	4
71	1520	20	53	30	1	10	5	4
72	1520	23	61	30	1	12	5	4
73	1733	27	61	30	2	12	5	4
74	2079	27	74	35	2	12	5	5
75	2079	27	74	40	2	12	6	6
76	2079	30	88	48	3	12	8	6
77	2371	35	90	57	3	12	9	6
78	2845	42	106	65	3	12	9	7
79	3243	50	116	78	3	12	11	9
80	3697	57	122	82	4	12	11	10
81	3709	68	126	82	4	12	11	13
82	3709	68	137	82	5	12	12	14
83	3709	68	137	82	6	12	12	14
84	3709	68	137	82	6	12	14	14
85	3709	68	137	82	7	12	14	16
86	3709	68	142	82	8	12	14	20
87	3709	68	143	82	9	12	14	20
88	3709	69	148	82	11	12	14	22
89	3709	69	153	82	13	12	14	25
90	3709	69	161	82	15	12	14	31
91	3709	69	163	82	17	12	14	31
92	3709	69	171	82	19	12	14	37
93	3709	69	171	82	19	12	14	37
94	3709	69	173	82	19	12	14	39
95	3709	69	173	82	19	12	14	39
96	3709	69	175	82	19	12	14	39
97	3709	69	176	82	19	12	14	39
98	3709	69	178	82	19	12	14	39
99	3709	69	178	82	19	12	14	39
100	3709	69	179	82	19	12	14	39
101	3709	69	179	82	19	12	14	39
102	3709	69	179	82	19	12	14	39
103	3709	69	182	82	19	12	14	39
104	3709	69	185	82	19	12	14	39
105	3709	69	188	82	19	12	14	39
106	3709	69	188	82	19	12	14	39
107	3709	69	188	82	19	12	14	39
108	3709	69	188	82	19	12	14	39
109	3709	69	192	82	19	12	14	39
110	3709	69	192	82	19	12	14	39
111	3709	69	196	82	19	12	14	39
112	3709	69	200	82	19	12	14	39
113	3709	69	205	82	19	12	14	39
114	3709	69	210	82	19	12	14	39
115	3709	69	210	82	19	12	14	39
116	3709	69	219	82	19	12	14	39
117	3709	69	226	82	19	12	14	39
118	3709	69	239	82	19	12	14	39
119	3709	69	249	82	19	12	14	39
120	3709	69	249	82	19	12	14	39
121	3709	69	249	82	19	12	14	39
122	3709	69	259	82	19	12	14	39

	Metric #							
time step	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
62	1	20	1	164852	2.24	1.87	4180	40
63	1	24	1	164852	2.24	1.57	4180	46
64	1	24	1	164852	1.79	1.32	4180	55
65	2	29	1	131882	1.79	1.11	3511	66
66	2	33	1	110781	1.79	1.11	3511	75
67	2	39	1	110781	1.51	0.89	3511	75
68	2	39	2	110781	1.51	0.89	2949	90
69	2	40	2	110781	1.51	0.89	2478	100
70	2	45	2	93056	1.51	0.71	2081	100
71	2	53	2	93056	1.20	0.60	2081	100
72	2	61	3	74445	0.96	0.60	2081	100
73	2	61	3	62533	0.96	0.50	1748	100
74	2	74	3	62533	0.96	0.50	1468	100
75	2	74	3	62533	0.81	0.42	1233	100
76	3	88	4	52528	0.65	0.34	1233	100
77	3	90	4	42022	0.52	0.27	1233	100
78	3	106	4	35299	0.52	0.23	1036	100
79	3	116	5	28239	0.44	0.23	1036	100
80	4	122	5	23721	0.44	0.23	870	100
81	5	126	5	18977	0.37	0.19	870	100
82	5	137	7	15940	0.37	0.19	696	100
83	5	137	7	15940	0.29	0.16	557	100
84	5	137	7	12752	0.29	0.13	446	100
85	5	137	9	12752	0.29	0.11	356	100
86	6	142	11	12752	0.23	0.11	299	100
87	6	143	11	10712	0.23	0.09	252	100
88	6	148	11	8570	0.23	0.07	252	100
89	6	153	13	8570	0.20	0.06	201	100
90	7	161	13	7198	0.20	0.06	201	100
91	7	163	15	6047	0.16	0.05	161	100
92	7	171	19	6047	0.13	0.04	135	100
93	7	171	19	5079	0.11	0.04	135	100
94	7	173	21	4267	0.09	0.04	135	100
95	7	173	25	4267	0.08	0.03	114	100
96	9	175	25	3413	0.07	0.03	91	100
97	10	176	25	2731	0.05	0.03	76	100
98	12	178	30	2294	0.04	0.02	61	100
99	12	178	30	2294	0.04	0.02	61	100
100	13	179	30	1835	0.04	0.02	49	100
101	13	179	30	1541	0.04	0.02	49	100
102	13	179	30	1541	0.03	0.02	39	100
103	16	182	30	1295	0.02	0.01	31	100
104	19	185	30	1088	0.02	0.01	26	100
105	22	188	30	914	0.02	0.01	21	100
106	22	188	30	914	0.02	0.01	17	100
107	22	188	30	731	0.01	0.01	17	100
108	22	188	30	731	0.01	0.01	14	100
109	26	192	30	731	0.01	0.01	12	100
110	26	192	30	731	0.01	0.01	12	100
111	30	196	30	731	0.01	0.00	9	100
112	34	200	30	731	0.01	0.00	8	100
113	39	205	30	731	0.01	0.00	8	100
114	44	210	30	585	0.00	0.00	8	100
115	44	210	30	491	0.00	0.00	6	100
116	53	219	30	491	0.00	0.00	5	100
117	60	226	30	491	0.00	0.00	5	100
118	73	239	30	491	0.00	0.00	4	100
119	83	249	30	393	0.00	0.00	3	100
120	83	249	30	393	0.00	0.00	2	100
121	83	249	30	393	0.00	0.00	2	100
122	93	259	30	330	0.00	0.00	2	100

C.9. Parameters for Network Trained on Progressively Stable-Unstable Data

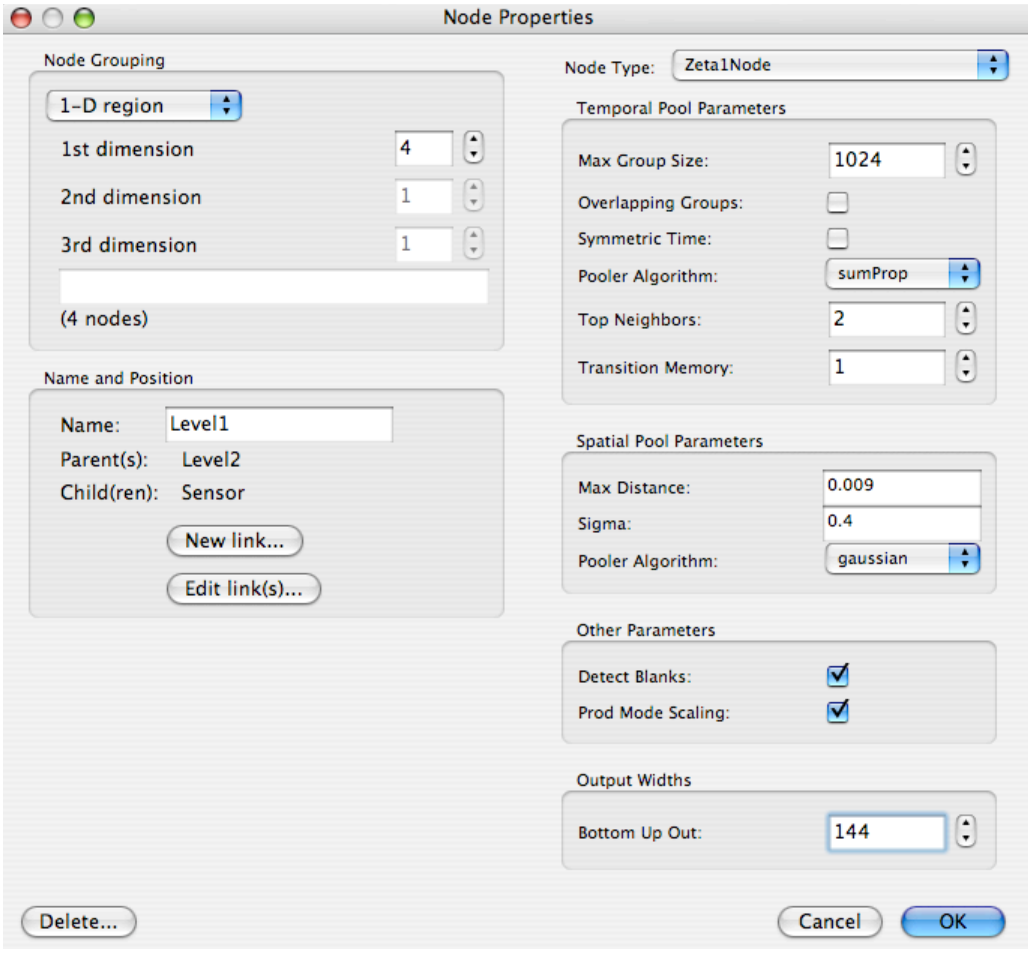


Figure 114: Bottom-Level Node Settings

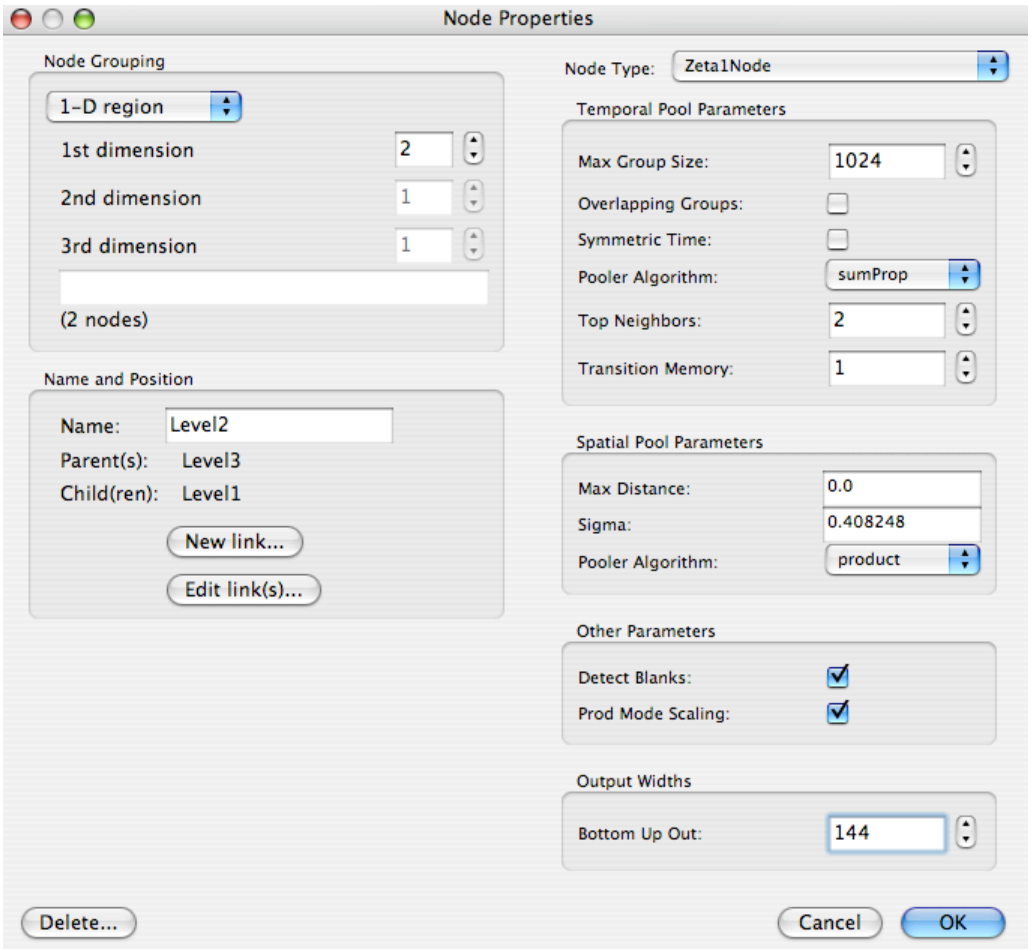


Figure 115: Second-Level Node Settings

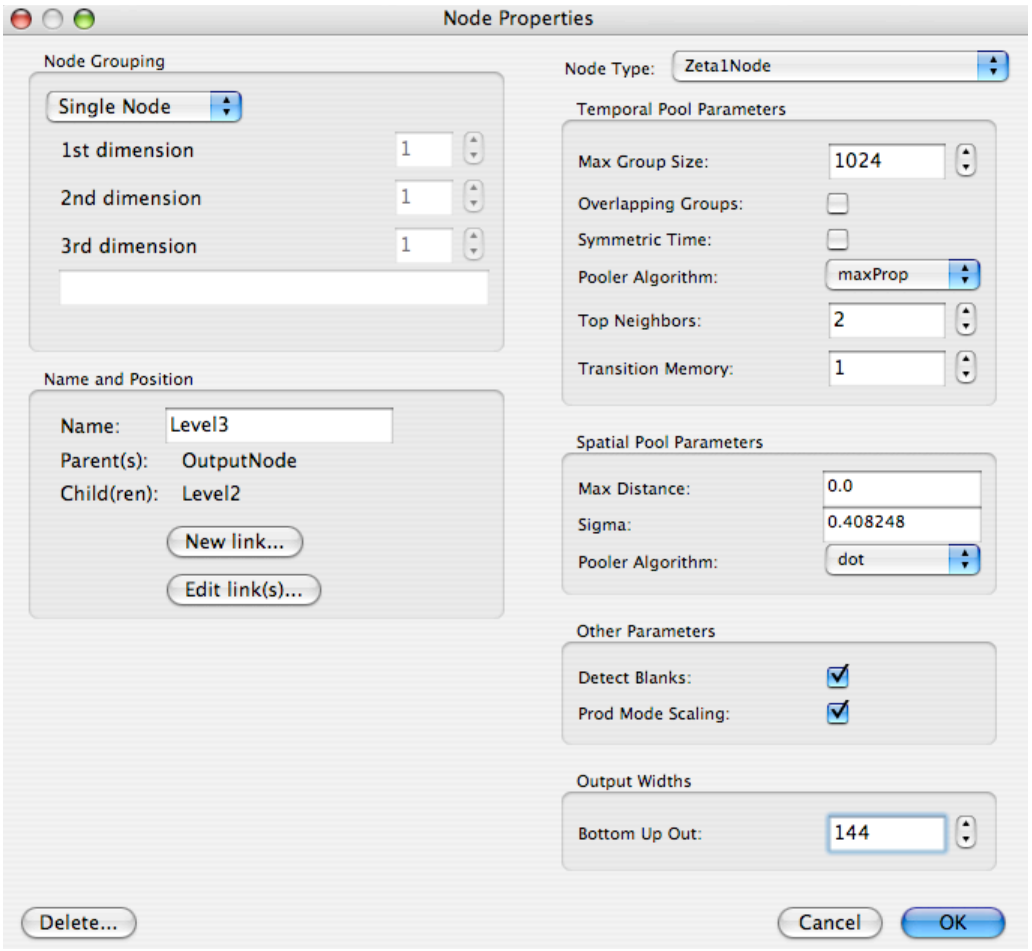


Figure 116: Top-Level Node Settings

C.10. Inference History on Progressively Stable-Unstable Data (Trained on it)

Time	First	Second
0	Group 0 1.530643	Group 2 1.367813
1	Group 0 1.554414	Group 2 1.389292
2	Group 0 1.613826	Group 2 1.433449
3	Group 0 1.620637	Group 2 1.449643
4	Group 0 1.66082	Group 2 1.499483
5	Group 0 1.670117	Group 2 1.507866
6	Group 0 1.717928	Group 2 1.564894
7	Group 0 1.751554	Group 2 1.604026
8	Group 0 1.764135	Group 2 1.622093
9	Group 0 1.788079	Group 2 1.655517
10	Group 0 1.791101	Group 2 1.668067
11	Group 0 1.811972	Group 2 1.693434
12	Group 0 1.888553	Group 2 1.779959
13	Group 0 1.898076	Group 2 1.786603
14	Group 0 1.926779	Group 2 1.823749
15	Group 0 1.927664	Group 2 1.832213
16	Group 0 1.931557	Group 2 1.83918
17	Group 0 1.952612	Group 2 1.860989
18	Group 0 1.964431	Group 2 1.86383

Time	First	Second
19	Group 0 1.957246	Group 2 1.86219
20	Group 0 1.966591	Group 2 1.878912
21	Group 0 1.97343	Group 2 1.885481
22	Group 0 1.975032	Group 2 1.891166
23	Group 0 1.971337	Group 2 1.895459
24	Group 0 1.951202	Group 2 1.879085
25	Group 0 1.976363	Group 2 1.911178
26	Group 0 1.977732	Group 2 1.915933
27	Group 0 1.978657	Group 2 1.923345
28	Group 0 1.980414	Group 2 1.931304
29	Group 3 1.958919	Group 6 1.915959
30	Group 3 1.93079	Group 5 1.908273
31	Group 5 1.933511	Group 3 1.924091
32	Group 5 1.948325	Group 0 1.93913
33	Group 0 1.971021	Group 5 1.926016
34	Group 0 1.975399	Group 5 1.932176
35	Group 0 1.980289	Group 5 1.905628
36	Group 0 1.985657	Group 5 1.879939
37	Group 0 2	Group 5 1.822005
38	Group 0 1.988042	Group 5 1.910054

Time	First	Second
39	Group 0 1.989191	Group 5 1.880279
40	Group 0 1.979469	Group 5 1.929251
41	Group 3 1.987345	Group 6 1.978926
42	Group 3 2	Group 6 1.99585
43	Group 3 1.976835	Group 6 1.974283
44	Group 6 1.971325	Group 3 1.970003
45	Group 2 1.936459	Group 0 1.933164
46	Group 2 1.981861	Group 0 1.974846
47	Group 2 1.991712	Group 0 1.981703
48	Group 4 1.987146	Group 0 1.973805
49	Group 4 1.960692	Group 0 1.94417
50	Group 2 1.977338	Group 0 1.958339
51	Group 2 2	Group 0 1.985039
52	Group 2 1.97382	Group 0 1.961364
53	Group 2 1.987124	Group 0 1.973121
54	Group 2 1.982774	Group 0 1.965623
55	Group 2 1.982482	Group 0 1.962352
56	Group 2 1.98235	Group 0 1.959963
57	Group 2 1.949334	Group 0 1.924245

Time	First	Second
59	Group 4 2	Group 0 1.977428
60	Group 7 2	Group 4 1.91567
61	blank 2	Group 7 0
62	Group 0 1.948991	Group 5 1.862445
63	Group 0 1.926705	Group 5 1.824857
64	Group 0 1.827377	Group 5 1.689999
65	Group 0 1.615583	Group 2 1.393824
66	Group 0 1.486802	Group 3 1.289156
67	Group 0 1.501219	Group 2 1.251038
68	Group 0 1.263743	Group 3 0.985973
69	Group 0 1.141475	Group 3 0.86736
70	Group 0 1.082188	Group 3 0.786098
71	Group 0 0.720766	Group 3 0.425266
72	Group 0 0.580166	Group 3 0.255422
73	Group 0 0.580444	Group 3 0.211801
74	Group 0 0.619025	Group 3 0.191586
75	Group 0 0.604825	Group 3 0.147258
76	Group 0 0.655084	Group 1 0.165228
77	Group 0 0.664012	Group 1 0.287435

Time	First	Second
79	Group 0 1.022522	Group 1 0.693721
80	Group 0 1.076575	Group 1 0.757363
81	Group 0 1.080021	Group 1 0.767764
82	Group 0 1.108976	Group 1 0.80259
83	Group 0 1.115497	Group 1 0.808884
84	Group 0 1.113954	Group 1 0.809534
85	Group 0 1.127359	Group 1 0.819409
86	Group 0 1.161711	Group 1 0.849262
87	Group 0 1.165901	Group 1 0.854602
88	Group 0 1.178953	Group 1 0.869359
89	Group 0 1.22812	Group 1 0.907969
90	Group 0 1.256363	Group 1 0.940264
91	Group 0 1.320437	Group 1 0.986892
92	Group 0 1.429379	Group 1 1.074558
93	Group 0 1.428646	Group 1 1.074572
94	Group 0 1.383569	Group 1 1.082507
95	Group 0 1.333642	Group 1 1.082507
96	Group 0 1.334534	Group 1 1.091369
97	Group 0 1.335672	Group 1 1.10177

Time	First	Second
99	Group 0 1.309633	Group 1 1.119428
100	Group 0 1.312255	Group 1 1.135014
101	Group 0 1.312221	Group 1 1.135019
102	Group 0 1.312221	Group 1 1.135019
103	Group 0 1.316679	Group 1 1.162011
104	Group 0 1.322244	Group 1 1.197368
105	Group 0 1.327088	Group 1 1.229927
106	Group 0 1.327088	Group 1 1.229927
107	Group 0 1.32707	Group 1 1.229931
108	Group 0 1.32707	Group 1 1.229931
109	Group 0 1.335309	Group 1 1.289536
110	Group 0 1.335309	Group 1 1.289536
111	Group 1 1.347107	Group 0 1.342528
112	Group 1 1.422954	Group 0 1.351083
113	Group 1 1.525832	Group 0 1.361249
114	Group 1 1.665036	Group 0 1.370253
115	Group 1 1.665037	Group 0 1.370244
116	Group 1 1.772427	Group 0 1.305654
117	Group 1 1.861551	Group 0 1.249956
118	Group 1 1.983569	Group 0 1.16202
119	Group 1 1.890025	Group 0 1.088616
120	Group 1 1.890025	Group 0 1.088616
121	Group 1 1.890025	Group 0 1.088616
122	Group 1 1.822204	Group 0 1.048151

C.11. Inference on Actual Data (Trained on Progressively Stable-Unstable Data)

Time	First	Second	Third
0	Group 7 1.677514	Group 2 1.586204	Group 0 1.526612
1	Group 0 1.224857	Group 4 1.035763	Group 7 1.022173
2	Group 0 1.691301	Group 7 1.656679	Group 4 1.442435
3	Group 0 1.277755	Group 4 1.251832	Group 2 1.210828
4	Group 0 1.873417	Group 2 1.762089	Group 4 1.599245
5	Group 0 1.517632	Group 5 1.460812	Group 3 1.359087
6	Group 5 1.536353	Group 0 1.534006	Group 3 1.489165
7	Group 3 1.650042	Group 5 1.639602	Group 0 1.637854
8	Group 5 1.659402	Group 0 1.649887	Group 3 1.640696
9	Group 5 1.763319	Group 0 1.749684	Group 3 1.715608
10	Group 0 1.615878	Group 5 1.603235	Group 3 1.543161
11	Group 5 0.81484	Group 0 0.798091	Group 3 0.694751
12	Group 0 1.1284	Group 5 1.080914	Group 3 0.939211
13	Group 0 1.660396	Group 5 1.596873	Group 3 1.434334
14	Group 0 1.608481	Group 5 1.561608	Group 3 1.372282
15	Group 0 1.50084	Group 5 1.450441	Group 4 1.33734
16	Group 0 1.432238	Group 5 1.426844	Group 3 1.268058
17	Group 0 1.549529	Group 5 1.516159	Group 3 1.400053
18	Group 1 0.970106	Group 0 0.584447	Group 5 0.520036
19	Group 0 1.521603	Group 5 1.457393	Group 3 1.357284
20	Group 0 1.280704	Group 5 1.224197	Group 3 1.101462
21	Group 0 1.571598	Group 5 1.450876	Group 2 1.364958
22	Group 0 1.751925	Group 5 1.639285	Group 2 1.536159
23	Group 0 1.597954	Group 5 1.528044	Group 3 1.463097
24	Group 0 1.288477	Group 5 1.249471	Group 3 1.132404
25	Group 0 1.319229	Group 5 1.295745	Group 3 1.146776
26	Group 0 1.458253	Group 5 1.410741	Group 3 1.249401
27	Group 0 1.220848	Group 5 1.197572	Group 3 1.057044
28	Group 0 1.389602	Group 5 1.348514	Group 3 1.224846
29	Group 0 1.070591	Group 2 0.992794	Group 5 0.866853
30	Group 2 1.251272	Group 0 1.242786	Group 4 1.204491
31	Group 0 1.338459	Group 4 1.275535	Group 2 1.266494
32	Group 0 1.456755	Group 2 1.442992	Group 4 1.335728
33	Group 0 1.319299	Group 2 1.311745	Group 5 1.296034
34	Group 0 1.507252	Group 5 1.476995	Group 3 1.373285
35	Group 0 1.241501	Group 5 1.240833	Group 3 1.171829
36	Group 5 1.038093	Group 0 1.035214	Group 6 0.954449
37	Group 5 0.994462	Group 0 0.972981	Group 6 0.887164
38	Group 0 1.292624	Group 5 1.290898	Group 6 1.186033
39	Group 0 1.232283	Group 5 1.212126	Group 6 1.101168
40	Group 0 0.949574	Group 5 0.928046	Group 3 0.81694
41	Group 0 1.143582	Group 1 0.939626	Group 5 0.745895
42	Group 0 1.043969	Group 5 0.816105	Group 1 0.748673
43	Group 0 1.28782	Group 1 0.980467	Group 5 0.719806
44	Group 0 1.051128	Group 2 0.963014	Group 4 0.768789
45	Group 5 1.074453	Group 0 1.062322	Group 3 1.006933
46	Group 0 0.971268	Group 5 0.927499	Group 3 0.856994
47	Group 0 1.244496	Group 1 0.991843	Group 5 0.733314
48	Group 0 1.282065	Group 1 1.047718	Group 5 0.692727
49	Group 0 1.168078	Group 5 0.947658	Group 1 0.869668
50	Group 0 1.195496	Group 5 1.132454	Group 3 0.962346
51	Group 0 1.405071	Group 5 1.305938	Group 3 1.1083
52	Group 0 1.570794	Group 5 1.429182	Group 3 1.212747
53	Group 0 1.702711	Group 5 1.54508	Group 3 1.348898
54	Group 0 1.735082	Group 5 1.620244	Group 6 1.566645
55	Group 0 1.847655	Group 5 1.744634	Group 6 1.711151
56	Group 0 1.703476	Group 5 1.623169	Group 6 1.560547
57	Group 0 1.786533	Group 5 1.713559	Group 6 1.683915
58	Group 0 1.661455	Group 5 1.599669	Group 3 1.50862
59	Group 0 1.967354	Group 5 1.875009	Group 3 1.788583

C.12. Inference History on Progressively Stable-Unstable Data

(Reduced *maxDistance* in Bottom-Level Nodes)

Time	First	Second
0	Group 0 1.835011	Group 4 1.350328
1	Group 0 1.866957	Group 4 1.367951
2	Group 0 1.909651	Group 4 1.4167
3	Group 0 1.940717	Group 4 1.40373
4	Group 0 1.94739	Group 4 1.445057
5	Group 0 1.953647	Group 4 1.439047
6	Group 0 1.962093	Group 4 1.492193
7	Group 0 1.968491	Group 4 1.540465
8	Group 0 1.976301	Group 4 1.528727
9	Group 0 1.972427	Group 4 1.585747
10	Group 0 1.932029	Group 4 1.614239
11	Group 0 1.875717	Group 4 1.675349
12	Group 0 1.9297	Group 4 1.692767
13	Group 0 1.896193	Group 4 1.652866
14	Group 0 1.886315	Group 4 1.688826
15	Group 0 1.889912	Group 4 1.764122
16	Group 0 1.915339	Group 4 1.832901
17	Group 0 1.928589	Group 4 1.823439
18	Group 0 1.926492	Group 4 1.860282
19	Group 0 1.912585	Group 4 1.898907
20	Group 0 1.913317	Group 4 1.911288
21	Group 4 1.931381	Group 9 1.900813
22	Group 4 1.957441	Group 9 1.93288
23	Group 4 1.930139	Group 9 1.906745
24	Group 4 1.952786	Group 5 1.928226
25	Group 4 1.895445	Group 5 1.854054
26	Group 4 1.89402	Group 5 1.87763
27	Group 5 1.90469	Group 4 1.883499
28	Group 5 1.874307	Group 2 1.824705
29	Group 5 1.831715	Group 1 1.776081
30	Group 5 1.838046	Group 2 1.78776
31	Group 5 1.855912	Group 1 1.854539
32	Group 1 1.914333	Group 5 1.847823
33	Group 1 1.954313	Group 8 1.795617
34	Group 1 2	Group 8 1.966878
35	Group 1 1.965677	Group 8 1.938104
36	Group 1 1.941999	Group 8 1.902058
37	Group 1 2	Group 8 1.717138
38	Group 1 1.971052	Group 8 1.80651
39	Group 1 1.971344	Group 8 1.798047
40	Group 1 1.988744	Group 8 1.976303
41	Group 1 1.973181	Group 8 1.965349
42	Group 1 1.903217	Group 8 1.89975
43	Group 5 1.825203	Group 1 1.819864
44	Group 8 1.910683	Group 1 1.908504
45	Group 2 1.902672	Group 5 1.898432
46	Group 2 1.97745	Group 9 1.969421
47	Group 2 1.992032	Group 9 1.980951
48	Group 2 2	Group 9 1.985524
49	Group 6 2	Group 2 1.988484
50	Group 2 1.984408	Group 4 1.964168
51	Group 2 1.980225	Group 9 1.964211
52	Group 2 2	Group 4 1.98671
53	Group 2 2	Group 4 1.985172
54	Group 2 1.995385	Group 6 1.984091
55	Group 2 1.992337	Group 5 1.971238
56	Group 2 1.980878	Group 5 1.957463
57	Group 2 1.944936	Group 5 1.918757
58	Group 6 1.984352	Group 2 1.974956
59	Group 7 1.943229	Group 2 1.94112
60	Group 7 1.852946	Group 2 1.787953
61	blank 2	Group 9 0
62	Group 1 1.922384	Group 8 1.71901
63	Group 1 1.894855	Group 5 1.729168
64	Group 9 1.680094	Group 5 1.659869
65	Group 0 1.741533	Group 9 1.559716
66	Group 0 1.668952	Group 5 1.438684
67	Group 0 1.696982	Group 9 1.360834
68	Group 0 1.67813	Group 9 1.119738
69	Group 0 1.623855	Group 4 0.984016
70	Group 0 1.408745	Group 4 0.876063
71	Group 0 0.94307	Group 4 0.509495
72	Group 0 0.662691	Group 4 0.32006
73	Group 0 0.582075	Group 4 0.259476
74	Group 0 0.630112	Group 4 0.232294
75	Group 0 0.641	Group 4 0.175229
76	Group 0 0.709482	Group 3 0.159168
77	Group 0 0.739257	Group 3 0.284028
78	Group 0 0.95276	Group 3 0.516014
79	Group 0 1.113111	Group 3 0.696522
80	Group 0 1.147953	Group 3 0.743295
81	Group 0 1.14664	Group 3 0.752731
82	Group 0 1.17278	Group 3 0.783294
83	Group 0 1.175298	Group 3 0.786525
84	Group 0 1.171882	Group 3 0.786837
85	Group 0 1.185309	Group 3 0.791556
86	Group 0 1.22024	Group 3 0.812402
87	Group 0 1.223983	Group 3 0.817086
88	Group 0 1.237176	Group 3 0.830475
89	Group 0 1.28882	Group 3 0.855939
90	Group 0 1.317951	Group 3 0.88435
91	Group 0 1.394645	Group 3 0.911188
92	Group 0 1.642739	Group 3 0.968744
93	Group 0 1.642765	Group 3 0.968755
94	Group 0 1.651456	Group 3 0.976454
95	Group 0 1.651456	Group 3 0.976454
96	Group 0 1.663128	Group 3 0.984428
97	Group 0 1.676574	Group 3 0.993839
98	Group 0 1.69879	Group 3 1.009943
99	Group 0 1.69879	Group 3 1.009943
100	Group 0 1.717784	Group 3 1.024285
101	Group 0 1.717792	Group 3 1.024289
102	Group 0 1.717792	Group 3 1.024289
103	Group 0 1.749371	Group 3 1.049404
104	Group 0 1.788373	Group 3 1.082806
105	Group 0 1.822106	Group 3 1.114038
106	Group 0 1.822106	Group 3 1.114038
107	Group 0 1.822112	Group 3 1.114042
108	Group 0 1.822112	Group 3 1.114042
109	Group 0 1.879014	Group 3 1.172279
110	Group 0 1.879014	Group 3 1.172279
111	Group 0 1.921915	Group 3 1.227584
112	Group 0 1.934114	Group 3 1.287785
113	Group 0 1.948219	Group 3 1.367415
114	Group 0 1.964558	Group 3 1.475715
115	Group 0 1.964558	Group 3 1.475718
116	Group 0 1.991771	Group 3 1.704638
117	Group 0 1.861488	Group 3 1.818857
118	Group 3 1.936857	Group 0 1.608384
119	Group 3 1.977314	Group 0 1.41851
120	Group 3 1.977314	Group 0 1.418509
121	Group 3 1.977314	Group 0 1.41851
122	Group 3 1.931052	Group 0 1.259334

C.13. Parameters for Network Trained on Progressively Stable-Unstable $N = 8$ Data

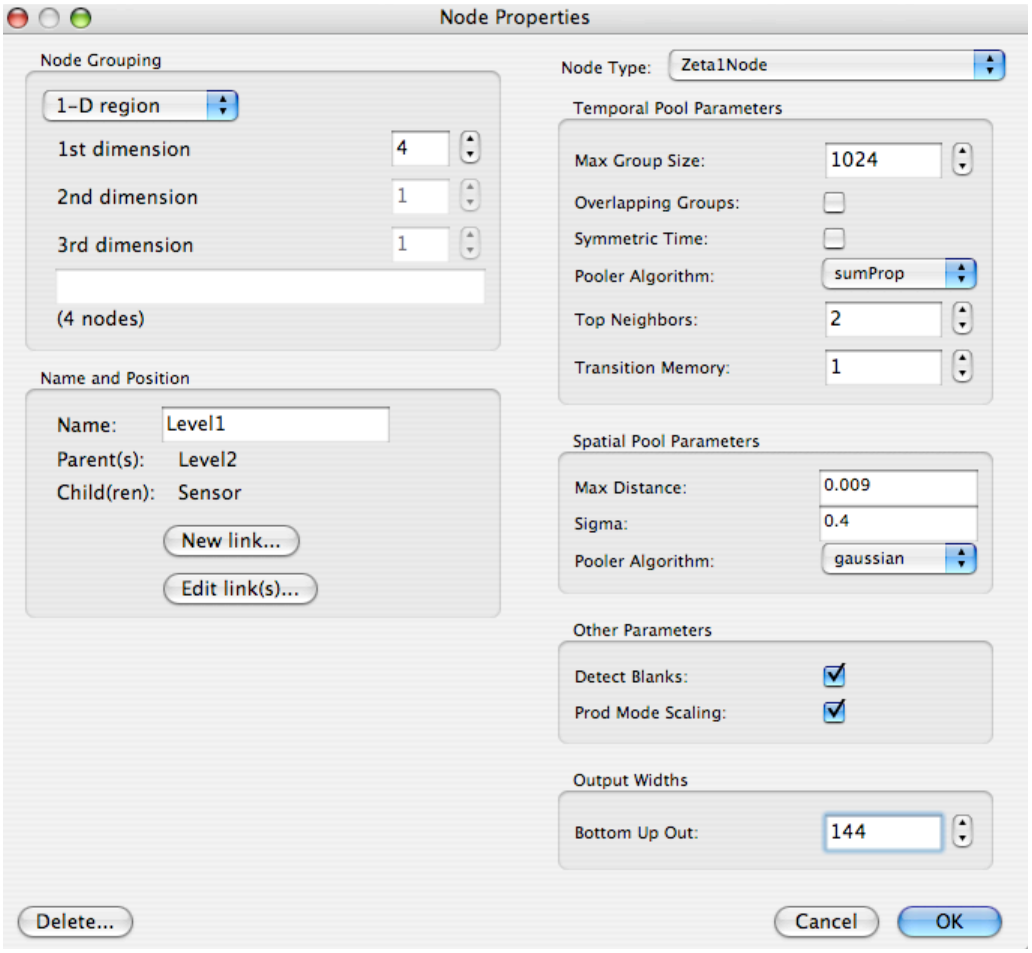


Figure 117: Bottom-Level Node Settings

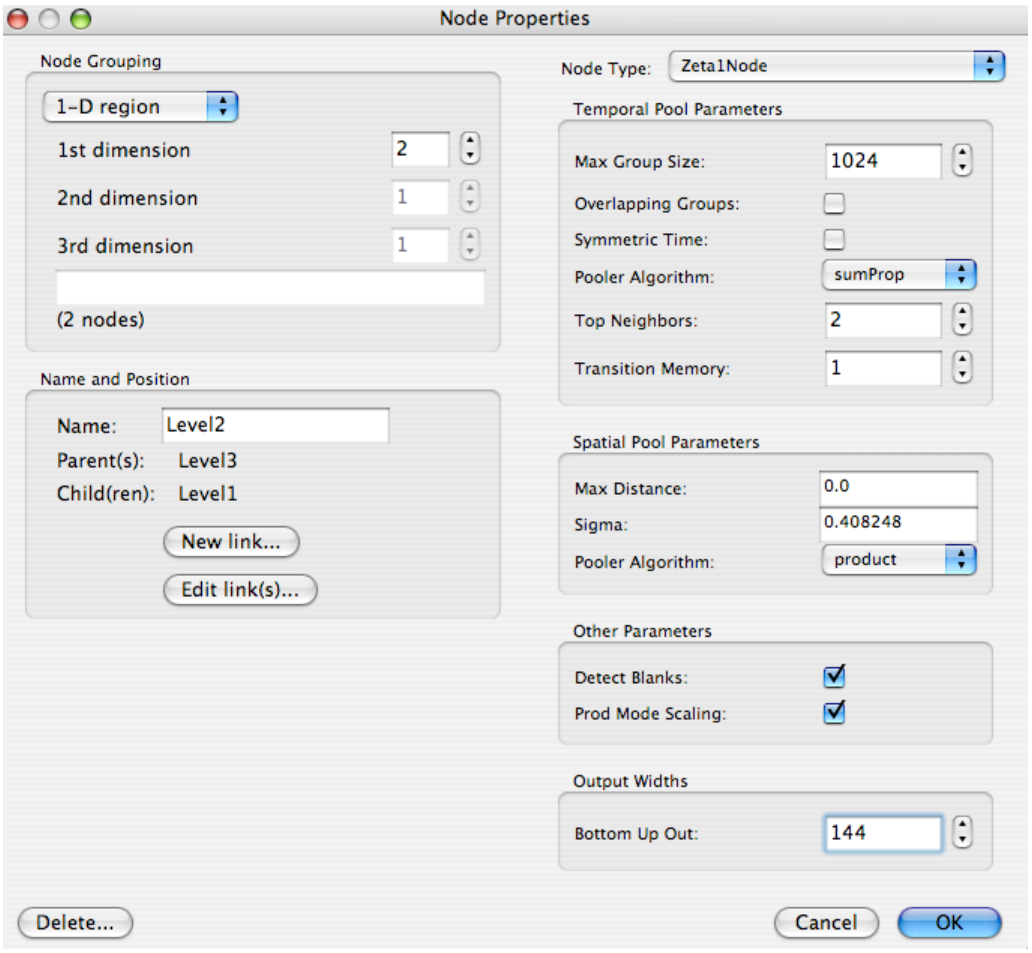


Figure 118: Second-Level Node Settings

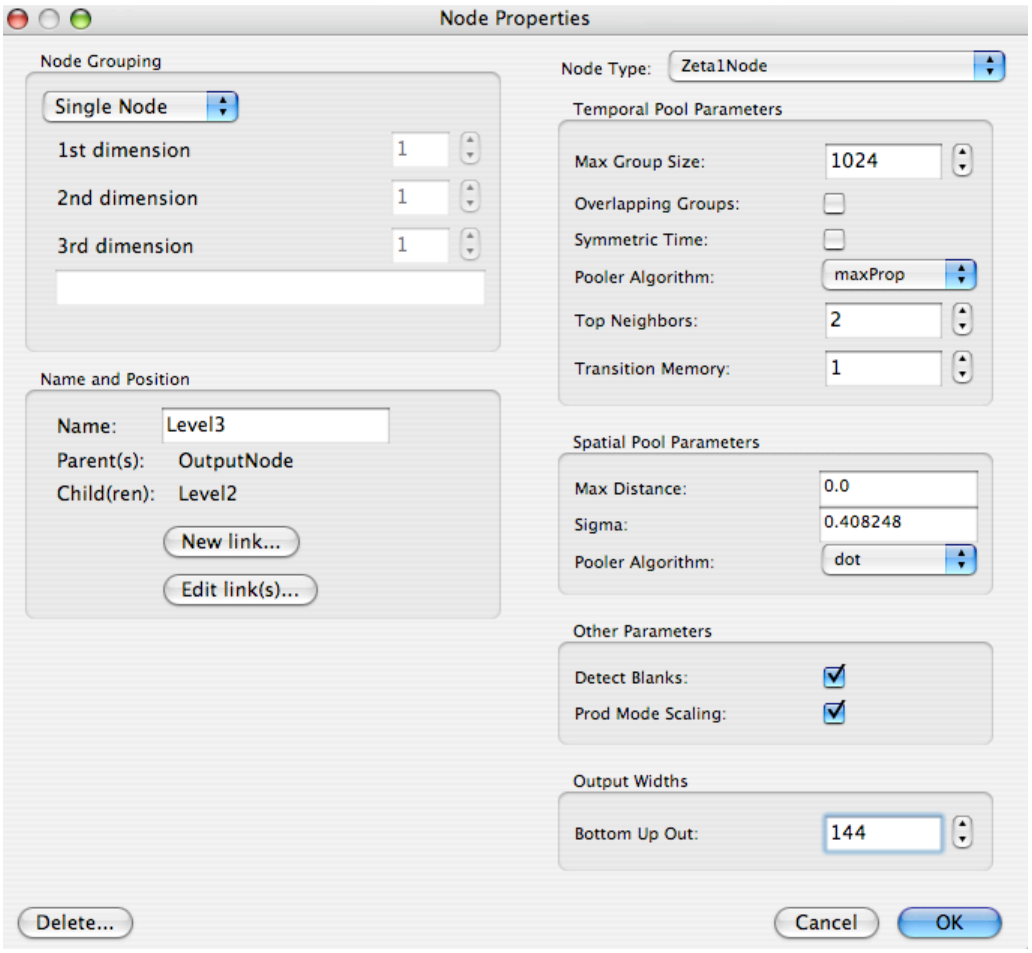


Figure 119: Top-Level Node Settings

C.14. Inference on $N = 8$ Actual Data (Trained on Progressively Stable-Unstable)

Time	First	Second	Third
0	Group 0 1.245419	Group 1 1.217586	Group 5 1.163105
1	Group 1 1.263426	Group 0 1.232642	Group 5 1.126925
2	Group 5 1.699682	Group 0 1.550892	Group 2 1.525483
3	Group 0 1.458653	Group 1 1.323049	Group 3 1.295542
4	Group 5 1.996714	Group 2 1.887566	Group 0 1.847262
5	Group 0 1.842915	Group 3 1.623966	Group 5 1.568197
6	Group 5 1.961665	Group 0 1.909919	Group 2 1.620571
7	Group 0 1.675914	Group 3 1.447612	Group 2 1.39554
8	Group 0 1.990315	Group 2 1.785194	Group 4 1.696778
9	Group 0 1.974777	Group 2 1.838994	Group 4 1.782876
10	Group 0 1.985157	Group 2 1.731766	Group 4 1.620034
11	Group 0 1.929858	Group 5 1.657662	Group 2 1.405655
12	Group 0 1.915269	Group 5 1.75401	Group 3 1.548014
13	Group 0 1.917147	Group 5 1.703914	Group 2 1.642719
14	Group 0 1.912885	Group 5 1.743298	Group 2 1.609206
15	Group 0 1.791347	Group 5 1.692056	Group 2 1.446047
16	Group 0 1.685408	Group 2 1.373089	Group 5 1.36327
17	Group 0 1.937545	Group 5 1.616742	Group 2 1.590205
18	Group 1 1.401518	Group 0 0.898207	Group 3 0.893176
19	Group 0 1.879291	Group 5 1.658879	Group 3 1.486474
20	Group 0 1.505068	Group 5 1.302968	Group 3 1.179821
21	Group 0 1.835732	Group 5 1.688977	Group 3 1.490469
22	Group 0 1.883967	Group 5 1.723829	Group 3 1.657484
23	Group 0 1.903989	Group 5 1.619691	Group 3 1.608001
24	Group 0 1.72256	Group 5 1.45674	Group 3 1.353086
25	Group 0 1.882064	Group 5 1.688096	Group 2 1.443145
26	Group 0 1.878654	Group 5 1.732586	Group 2 1.53188
27	Group 0 1.202928	Group 2 1.062618	Group 5 1.05184
28	Group 0 1.508504	Group 5 1.409273	Group 2 1.25322
29	Group 0 1.627999	Group 5 1.537677	Group 3 1.236575
30	Group 0 1.777073	Group 5 1.661485	Group 1 1.343653
31	Group 0 1.804516	Group 5 1.67693	Group 3 1.45839
32	Group 0 1.889057	Group 5 1.725105	Group 3 1.499619
33	Group 0 1.689835	Group 5 1.446861	Group 3 1.380333
34	Group 0 1.610203	Group 5 1.42049	Group 3 1.365714
35	Group 0 1.48362	Group 3 1.215937	Group 5 1.179466
36	Group 0 1.066529	Group 1 1.056793	Group 3 1.021708
37	Group 0 1.068825	Group 2 1.032038	Group 4 0.946209
38	Group 0 1.043429	Group 2 1.017839	Group 4 0.925146
39	Group 0 1.074599	Group 2 1.014527	Group 5 0.907256
40	Group 0 0.988666	Group 2 0.966204	Group 1 0.932073
41	Group 0 0.962701	Group 2 0.948633	Group 1 0.870167
42	Group 1 1.020232	Group 0 0.996148	Group 3 0.985646
43	Group 0 1.017981	Group 3 1.005095	Group 1 0.947405
44	Group 0 1.089154	Group 3 1.03512	Group 1 0.967625
45	Group 0 1.026258	Group 3 0.972702	Group 1 0.954098
46	Group 0 1.25413	Group 3 1.113858	Group 5 1.067977
47	Group 1 1.321889	Group 0 0.968131	Group 3 0.951316
48	Group 1 1.374881	Group 0 1.014861	Group 3 0.974524
49	Group 1 1.171198	Group 0 1.137292	Group 5 1.004322
50	Group 0 1.120691	Group 5 1.063956	Group 1 1.029627
51	Group 0 1.771086	Group 5 1.711737	Group 1 1.357411
52	Group 0 1.814709	Group 5 1.75024	Group 2 1.46198
53	Group 0 1.806045	Group 5 1.715735	Group 2 1.531337
54	Group 0 1.872122	Group 2 1.656495	Group 5 1.637165
55	Group 0 1.888082	Group 2 1.722708	Group 5 1.687913
56	Group 0 1.876426	Group 5 1.70812	Group 2 1.646058
57	Group 0 1.905865	Group 2 1.723396	Group 5 1.696307
58	Group 0 1.901126	Group 5 1.70251	Group 2 1.656572
59	Group 0 1.893539	Group 2 1.856601	Group 3 1.775081

C.15. Inference on Progressively Stable-Unstable Training Data

(Permutation #3)

Time	First	Second
0	Group 0 1.750698	Group 4 1.684764
1	Group 0 1.787816	Group 4 1.722524
2	Group 0 1.801716	Group 4 1.746328
3	Group 0 1.861522	Group 4 1.789361
4	Group 0 1.875026	Group 4 1.81913
5	Group 0 1.879743	Group 4 1.831223
6	Group 0 1.897289	Group 4 1.866392
7	Group 0 1.924618	Group 4 1.902809
8	Group 0 1.933147	Group 4 1.920768
9	Group 0 1.945873	Group 4 1.941957
10	Group 4 1.952325	Group 0 1.918354
11	Group 4 1.942105	Group 0 1.911085
12	Group 4 1.980206	Group 5 1.943954
13	Group 4 1.958489	Group 5 1.894131
14	Group 4 1.948882	Group 5 1.895366
15	Group 5 1.929108	Group 4 1.924103
16	Group 5 1.927648	Group 4 1.917563
17	Group 4 1.953087	Group 5 1.94091
18	Group 4 1.904866	Group 8 1.890084

Time	First	Second
19	Group 8 1.87131	Group 0 1.866547
20	Group 9 1.907427	Group 0 1.906531
21	Group 0 1.939294	Group 9 1.917197
22	Group 0 1.942997	Group 9 1.913735
23	Group 0 1.95687	Group 9 1.887684
24	Group 6 1.981452	Group 0 1.929836
25	Group 6 1.932565	Group 0 1.905756
26	Group 0 1.943134	Group 6 1.936954
27	Group 0 1.968946	Group 6 1.946238
28	Group 0 1.957548	Group 6 1.882841
29	Group 0 1.963815	Group 6 1.932405
30	Group 0 1.966032	Group 6 1.935951
31	Group 0 1.98764	Group 6 1.906933
32	Group 0 1.9855	Group 1 1.886405
33	Group 0 1.973827	Group 1 1.825836
34	Group 0 1.957646	Group 6 1.763065
35	Group 0 1.94028	Group 6 1.702318
36	Group 0 1.916091	Group 1 1.636889
37	Group 0 1.902117	Group 1 1.787713
38	Group 0 1.961398	Group 1 1.793502

Time	First	Second
39	Group 0 1.954789	Group 1 1.757278
40	Group 0 1.946571	Group 1 1.705249
41	Group 0 1.978666	Group 1 1.792315
42	Group 0 1.917678	Group 6 1.795104
43	Group 0 1.93291	Group 6 1.910817
44	Group 0 1.941719	Group 6 1.871871
45	Group 1 1.979948	Group 6 1.978452
46	Group 1 1.984315	Group 6 1.962348
47	Group 1 1.971804	Group 7 1.931793
48	Group 1 1.945788	Group 7 1.92542
49	Group 1 1.938066	Group 9 1.921381
50	Group 1 1.968621	Group 6 1.890745
51	Group 1 1.982563	Group 0 1.884851
52	Group 1 1.967312	Group 0 1.961329
53	Group 1 1.967928	Group 0 1.952971
54	Group 1 1.997244	Group 0 1.957446
55	Group 1 1.992135	Group 0 1.927963
56	Group 1 1.975674	Group 0 1.911267
57	Group 1 1.95714	Group 0 1.918329
58	Group 1 1.955797	Group 0 1.884216

Time	First	Second
59	Group 7 1.950028	Group 1 1.897765
60	Group 7 1.980613	Group 1 1.828173
61	blank 2	Group 9 0
62	Group 0 1.865213	Group 6 1.630333
63	Group 0 1.846963	Group 6 1.659314
64	Group 0 1.796771	Group 6 1.679208
65	Group 0 1.589455	Group 8 1.568061
66	Group 0 1.550896	Group 8 1.412768
67	Group 0 1.465587	Group 4 1.409067
68	Group 0 1.347204	Group 4 1.217448
69	Group 0 1.243076	Group 4 1.069988
70	Group 0 1.133652	Group 4 0.940643
71	Group 0 1.166666	Group 4 0.851114
72	Group 0 1.139832	Group 4 0.702933
73	Group 0 1.140408	Group 4 0.638427
74	Group 0 1.158698	Group 4 0.584592
75	Group 0 1.21899	Group 2 0.568174
76	Group 0 1.362926	Group 2 0.77878
77	Group 0 1.286039	Group 2 0.708322
78	Group 0 1.028974	Group 2 0.697051

Time	First	Second
79	Group 0 0.931858	Group 2 0.695319
80	Group 0 0.894759	Group 2 0.709534
81	Group 0 0.879223	Group 2 0.736803
82	Group 0 0.892579	Group 2 0.780082
83	Group 0 0.901683	Group 2 0.803494
84	Group 0 0.902331	Group 2 0.806182
85	Group 0 0.945085	Group 2 0.870685
86	Group 0 1.111995	Group 2 0.996115
87	Group 0 1.243189	Group 2 1.093534
88	Group 0 1.484244	Group 2 1.274698
89	Group 0 1.53916	Group 2 1.319007
90	Group 0 1.553992	Group 2 1.343971
91	Group 0 1.555696	Group 2 1.346893
92	Group 0 1.562714	Group 2 1.358931
93	Group 0 1.562704	Group 2 1.358922
94	Group 0 1.564714	Group 2 1.362391
95	Group 0 1.564703	Group 2 1.36238
96	Group 0 1.572524	Group 2 1.375959
97	Group 0 1.581521	Group 2 1.391762

Time	First	Second
98	Group 0 1.596375	Group 2 1.41827
99	Group 0 1.596375	Group 2 1.41827
100	Group 0 1.609043	Group 2 1.441294
101	Group 0 1.609039	Group 2 1.44129
102	Group 0 1.609036	Group 2 1.441287
103	Group 0 1.630039	Group 2 1.480312
104	Group 0 1.655866	Group 2 1.529765
105	Group 0 1.671803	Group 2 1.567996
106	Group 0 1.671802	Group 2 1.567994
107	Group 0 1.6718	Group 2 1.567992
108	Group 0 1.671799	Group 2 1.567991
109	Group 0 1.671798	Group 2 1.609084
110	Group 0 1.671797	Group 2 1.609083
111	Group 0 1.671797	Group 2 1.645054
112	Group 2 1.671796	Group 0 1.656096
113	Group 2 1.671796	Group 0 1.609081
114	Group 2 1.671795	Group 0 1.558217
115	Group 2 1.671794	Group 0 1.558216
116	Group 2 1.649528	Group 0 1.463198
117	Group 2 1.558	Group 3 1.485899
118	Group 3 1.56459	Group 2 1.295089
119	Group 3 1.571348	Group 2 1.090859
120	Group 3 1.571348	Group 2 1.090858
121	Group 3 1.571348	Group 2 1.090858
122	Group 3 1.499902	Group 2 0.92198

C.16. Inference on Actual Data (Permutation #3)

Time	First	Second	Third
0	Group 4 1.621942	Group 0 1.198094	Group 7 1.16488
1	Group 4 1.504226	Group 0 1.488513	Group 5 1.188406
2	Group 4 1.642349	Group 9 1.61289	Group 7 1.488207
3	Group 0 1.639534	Group 4 1.502653	Group 5 1.365874
4	Group 1 1.879492	Group 0 1.847068	Group 9 1.83371
5	Group 0 1.592732	Group 8 1.543983	Group 4 1.423173
6	Group 8 1.615058	Group 0 1.577355	Group 6 1.462074
7	Group 0 1.522382	Group 8 1.396874	Group 6 1.181409
8	Group 8 1.519224	Group 0 1.499767	Group 6 1.344644
9	Group 0 1.804873	Group 6 1.702427	Group 8 1.619938
10	Group 0 1.395168	Group 8 1.3145	Group 6 1.180085
11	Group 0 1.215444	Group 8 1.193372	Group 4 1.09787
12	Group 0 1.210828	Group 8 1.205968	Group 4 1.141308
13	Group 8 1.355855	Group 0 1.344333	Group 4 1.236309
14	Group 0 1.321706	Group 8 1.256847	Group 6 1.198007
15	Group 8 1.592997	Group 0 1.590691	Group 4 1.518069
16	Group 8 1.380648	Group 0 1.37905	Group 4 1.178422
17	Group 0 1.252419	Group 8 1.191684	Group 6 1.049701
18	Group 0 1.44259	Group 8 1.357377	Group 4 1.311746
19	Group 0 1.651351	Group 8 1.627468	Group 4 1.401373
20	Group 0 1.00969	Group 8 0.99848	Group 4 0.901295
21	Group 0 1.689408	Group 8 1.685116	Group 4 1.486131
22	Group 8 1.762258	Group 0 1.727901	Group 4 1.561555
23	Group 0 1.49608	Group 8 1.473364	Group 4 1.321162
24	Group 0 1.33559	Group 8 1.302834	Group 4 1.207474
25	Group 8 1.551584	Group 0 1.538705	Group 4 1.405604
26	Group 0 1.43272	Group 8 1.386875	Group 6 1.255477
27	Group 8 1.569454	Group 0 1.543832	Group 4 1.437428
28	Group 8 1.463972	Group 0 1.415686	Group 4 1.278728
29	Group 8 1.401138	Group 0 1.394636	Group 4 1.310448
30	Group 0 1.736417	Group 8 1.71545	Group 4 1.587031
31	Group 0 1.765499	Group 8 1.738769	Group 4 1.646115
32	Group 8 1.839558	Group 4 1.833888	Group 0 1.807391
33	Group 0 1.675207	Group 8 1.674814	Group 4 1.535206
34	Group 8 1.592423	Group 0 1.549667	Group 4 1.483529
35	Group 0 1.493042	Group 8 1.441577	Group 4 1.273998
36	Group 0 1.393062	Group 8 1.370991	Group 4 1.212755
37	Group 0 1.428944	Group 8 1.401417	Group 4 1.236272
39	Group 0 1.452435	Group 8 1.405104	Group 4 1.226689
40	Group 8 1.530595	Group 0 1.488437	Group 4 1.368191
41	Group 8 1.251087	Group 0 1.225974	Group 4 1.124162
42	Group 0 1.203259	Group 8 1.199741	Group 4 1.09114
43	Group 0 0.795118	Group 8 0.732738	Group 1 0.650269
44	Group 8 1.604814	Group 4 1.602947	Group 0 1.583802
45	Group 0 1.0237	Group 8 0.993871	Group 4 0.855723
46	Group 8 1.585737	Group 0 1.554032	Group 4 1.433297
47	Group 8 1.238294	Group 0 1.236427	Group 4 1.103186
48	Group 0 1.346937	Group 8 1.339711	Group 4 1.188213
49	Group 0 1.575781	Group 8 1.540752	Group 4 1.406754
50	Group 8 1.494062	Group 0 1.47073	Group 4 1.323453
51	Group 0 1.563786	Group 8 1.550962	Group 4 1.382916
52	Group 0 1.513512	Group 8 1.292672	Group 6 1.248586
53	Group 0 1.566527	Group 8 1.29377	Group 6 1.272879
54	Group 0 1.667522	Group 8 1.509258	Group 6 1.399943
55	Group 0 1.730338	Group 6 1.46275	Group 8 1.373185
56	Group 0 1.657027	Group 8 1.529712	Group 6 1.41717
57	Group 0 1.722559	Group 8 1.468362	Group 6 1.467062
58	Group 0 1.5486	Group 8 1.345184	Group 6 1.297285
59	Group 0 1.828819	Group 6 1.607076	Group 8 1.50799

C.17. Inference on Actual Data (Permutation #13)

Time	First	Second
0	Group 0 1.292203	Group 3 1.000692
1	Group 0 0.887469	Group 3 0.492849
2	Group 0 1.187918	Group 3 0.733627
3	Group 0 1.095309	Group 1 0.727858
4	Group 0 0.911765	Group 3 0.558822
5	Group 0 1.303543	Group 1 0.981143
6	Group 0 1.184456	Group 1 0.64377
7	Group 0 1.254915	Group 1 0.890781
8	Group 0 1.281935	Group 1 0.943789
9	Group 0 1.362513	Group 1 0.999392
10	Group 0 1.323664	Group 1 0.988434
11	Group 1 1.586376	Group 0 1.36133
12	Group 0 1.311341	Group 1 1.050112
13	Group 0 1.32683	Group 1 1.028068
14	Group 0 1.346789	Group 1 1.038291
15	Group 0 1.464955	Group 1 1.019713
16	Group 0 1.434745	Group 1 1.116362
17	Group 1 1.852864	Group 0 1.460853
18	Group 0 1.254081	Group 1 0.918479
19	Group 0 1.393278	Group 1 0.88307
20	Group 0 1.293581	Group 1 0.881698
21	Group 0 1.305655	Group 1 0.917284
22	Group 0 1.286039	Group 1 0.928482
23	Group 0 1.226403	Group 1 0.942707
24	Group 0 1.308463	Group 1 1.065565
25	Group 0 1.422005	Group 1 1.052766
26	Group 0 1.35358	Group 1 1.009196
27	Group 0 1.426149	Group 1 0.99557
28	Group 0 1.365992	Group 1 1.005433
29	Group 0 1.251767	Group 1 0.943974
30	Group 0 1.328177	Group 1 0.940599
31	Group 0 1.307804	Group 1 0.953046
32	Group 0 1.072524	Group 1 0.75146
33	Group 0 1.321415	Group 1 0.931864
34	Group 0 1.334876	Group 1 0.984621
35	Group 0 1.359222	Group 1 0.93266
36	Group 0 1.402523	Group 1 0.960765
37	Group 0 1.432078	Group 1 1.004449
38	Group 0 1.400836	Group 1 1.008046
39	Group 0 1.403999	Group 1 1.002983
40	Group 0 1.396332	Group 1 0.985331
41	Group 0 1.493045	Group 1 0.97873
42	Group 0 1.373842	Group 1 0.928824
43	Group 0 1.406193	Group 1 0.900024
44	Group 0 1.231097	Group 1 0.833262
45	Group 0 1.388687	Group 1 0.91445
46	Group 0 1.301415	Group 1 0.91593
47	Group 0 1.427075	Group 1 0.95149
48	Group 0 1.347744	Group 1 0.930882
49	Group 0 1.484158	Group 1 1.014226
50	Group 0 1.399254	Group 1 1.010518
51	Group 0 1.28242	Group 1 1.002836
52	Group 0 1.372083	Group 1 1.011786
53	Group 0 1.345332	Group 1 1.005491
54	Group 0 1.372366	Group 1 1.000356
55	Group 0 1.3727	Group 1 1.003596
56	Group 0 1.353096	Group 1 0.983642
57	Group 0 1.336591	Group 1 0.970599
58	Group 0 1.341078	Group 1 1.009172
59	Group 0 1.332742	Group 1 1.013189

C.18. Monotonically Stable-Unstable Data

	Metric #							
time step	1	2	3	4	5	6	7	8
0	0.20221084	0.11594203	0.14598540	0.13414634	0.05263158	0.25000000	0.14285714	0.02564103
1	0.19210030	0.11014493	0.13868613	0.12743902	0.05000000	0.23750000	0.13571429	0.02435897
2	0.18249528	0.10463768	0.13175182	0.12106707	0.04750000	0.22562500	0.12892857	0.02314103
3	0.17337052	0.09940580	0.12516423	0.11501372	0.04512500	0.21434375	0.12248214	0.02198397
4	0.16470199	0.09443551	0.11890602	0.10926303	0.04286875	0.20362656	0.11635804	0.02088478
5	0.15646689	0.08971373	0.11296072	0.10379988	0.04072531	0.19344523	0.11054013	0.01984054
6	0.14864355	0.08522805	0.10731268	0.09860989	0.03868905	0.18377297	0.10501313	0.01884851
7	0.14121137	0.08096664	0.10194705	0.09367939	0.03675459	0.17458432	0.09976247	0.01790608
8	0.13415080	0.07691831	0.09684970	0.08899542	0.03491686	0.16585511	0.09477435	0.01701078
9	0.12744326	0.07307240	0.09200721	0.08454565	0.03317102	0.15756235	0.09003563	0.01616024
10	0.12107110	0.06941878	0.08740685	0.08031837	0.03151247	0.14968423	0.08553385	0.01535223
11	0.11501754	0.06594784	0.08303651	0.07630245	0.02993685	0.14220002	0.08125716	0.01458462
12	0.10926667	0.06265044	0.07888468	0.07248733	0.02844000	0.13509002	0.07719430	0.01385539
13	0.10380333	0.05951792	0.07494045	0.06886296	0.02701800	0.12833552	0.07333458	0.01316262
14	0.09861317	0.05654203	0.07119343	0.06541981	0.02566710	0.12191874	0.06966785	0.01250449
15	0.09368251	0.05371493	0.06763376	0.06214882	0.02438375	0.11582281	0.06618446	0.01187926
16	0.08899838	0.05102918	0.06425207	0.05904138	0.02316456	0.11003167	0.06287524	0.01128530
17	0.08454846	0.04847772	0.06103946	0.05608931	0.02200633	0.10453008	0.05973148	0.01072103
18	0.08032104	0.04605383	0.05798749	0.05328485	0.02090602	0.09930358	0.05674490	0.01018498
19	0.07630499	0.04375114	0.05508812	0.05062061	0.01986072	0.09433840	0.05390766	0.00967573
20	0.07248974	0.04156359	0.05233371	0.04808957	0.01886768	0.08962148	0.05121227	0.00919195
21	0.06886525	0.03948541	0.04971703	0.04568510	0.01792430	0.08514041	0.04865166	0.00873235
22	0.06542199	0.03751114	0.04723117	0.04340084	0.01702808	0.08088339	0.04621908	0.00829573
23	0.06215089	0.03563558	0.04486962	0.04123080	0.01617668	0.07683922	0.04390812	0.00788095
24	0.05904335	0.03385380	0.04262613	0.03916926	0.01536784	0.07299726	0.04171272	0.00748690
25	0.05609118	0.03216111	0.04049483	0.03721080	0.01459945	0.06934739	0.03962708	0.00711255
26	0.05328662	0.03055305	0.03847009	0.03535026	0.01386948	0.06588002	0.03764573	0.00675693
27	0.05062229	0.02902540	0.03654658	0.03358274	0.01317600	0.06258602	0.03576344	0.00641908
28	0.04809117	0.02757413	0.03471925	0.03190361	0.01251720	0.05945672	0.03397527	0.00609813
29	0.04568662	0.02619543	0.03298329	0.03030843	0.01189134	0.05648389	0.03227651	0.00579322
30	0.04340228	0.02488565	0.03133413	0.02879300	0.01129678	0.05365969	0.03066268	0.00550356
31	0.04123217	0.02364137	0.02976742	0.02735335	0.01073194	0.05097671	0.02912955	0.00522838
32	0.03917056	0.02245930	0.02827905	0.02598569	0.01019534	0.04842787	0.02767307	0.00496696
33	0.03721203	0.02133634	0.02686510	0.02468640	0.00968557	0.04600648	0.02628942	0.00471861
34	0.03535143	0.02026952	0.02552184	0.02345208	0.00920130	0.04370615	0.02497494	0.00448268
35	0.03358386	0.01925604	0.02424575	0.02227948	0.00874123	0.04152085	0.02372620	0.00425855
36	0.03190467	0.01829324	0.02303346	0.02116550	0.00830417	0.03944480	0.02253989	0.00404562
37	0.03030943	0.01737858	0.02188179	0.02010723	0.00788896	0.03747256	0.02141289	0.00384334
38	0.02879396	0.01650965	0.02078770	0.01910187	0.00749451	0.03559894	0.02034225	0.00365117
39	0.02735426	0.01568417	0.01974831	0.01814677	0.00711979	0.03381899	0.01932514	0.00346861
40	0.02598655	0.01489996	0.01876090	0.01723944	0.00676380	0.03212804	0.01835888	0.00329518
41	0.02468722	0.01415496	0.01782285	0.01637746	0.00642561	0.03052164	0.01744094	0.00313042
42	0.02345286	0.01344721	0.01693171	0.01555859	0.00610433	0.02899556	0.01656889	0.00297390
43	0.02228022	0.01277485	0.01608513	0.01478066	0.00579911	0.02754578	0.01574044	0.00282521
44	0.02116621	0.01213611	0.01528087	0.01404163	0.00550916	0.02616849	0.01495342	0.00268395
45	0.02010790	0.01152931	0.01451683	0.01333955	0.00523370	0.02486006	0.01420575	0.00254975
46	0.01910250	0.01095284	0.01379098	0.01267257	0.00497201	0.02361706	0.01349546	0.00242226
47	0.01814738	0.01040520	0.01310144	0.01203894	0.00472341	0.02243621	0.01282069	0.00230115
48	0.01724001	0.00988494	0.01244636	0.01143699	0.00448724	0.02131440	0.01217966	0.00218609
49	0.01637801	0.00939069	0.01182405	0.01086514	0.00426288	0.02024868	0.01157067	0.00207679
50	0.01555911	0.00892116	0.01123284	0.01032189	0.00404974	0.01923624	0.01099214	0.00197295
51	0.01478115	0.00847510	0.01067120	0.00980579	0.00384725	0.01827443	0.01044253	0.00187430
52	0.01404209	0.00805134	0.01013764	0.00931550	0.00365489	0.01736071	0.00992041	0.00178059
53	0.01333999	0.00764878	0.00963076	0.00884973	0.00347214	0.01649267	0.00942439	0.00169156
54	0.01267299	0.00726634	0.00914922	0.00840724	0.00329853	0.01566804	0.00895317	0.00160698
55	0.01203934	0.00690302	0.00869176	0.00798688	0.00313361	0.01488464	0.00850551	0.00152663
56	0.01143737	0.00655787	0.00825717	0.00758754	0.00297693	0.01414041	0.00808023	0.00145030
57	0.01086551	0.00622998	0.00784431	0.00720816	0.00282808	0.01343339	0.00767622	0.00137778
58	0.01032223	0.00591848	0.00745210	0.00684775	0.00268668	0.01276172	0.00729241	0.00130889
59	0.00980612	0.00562255	0.00707949	0.00650536	0.00255234	0.01212363	0.00692779	0.00124345
60	0.00931581	0.00534143	0.00672552	0.00618009	0.00242473	0.01151745	0.00658140	0.00118128
61	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000

	Metric #							
time step	9	10	11	12	13	14	15	16
0	0.01075269	0.14598540	0.03333333	0.90083060	0.89101034	0.96891192	0.86008230	0.56666667
1	0.01021505	0.13868613	0.03166667	0.90533475	0.89546539	0.97375648	0.86438272	0.54400000
2	0.00970430	0.13175182	0.03008333	0.90986143	0.89994272	0.97862526	0.86870463	0.52224000
3	0.00921909	0.12516423	0.02857917	0.91441074	0.90444243	0.98351839	0.87304815	0.50135040
4	0.00875813	0.11890602	0.02715021	0.91898279	0.90896465	0.98843598	0.87741339	0.48129638
5	0.00832023	0.11296072	0.02579270	0.92357770	0.91350947	0.99337816	0.88180046	0.46204453
6	0.00790421	0.10731268	0.02450306	0.92819559	0.91807702	0.99834505	0.88620946	0.44356275
7	0.00750900	0.10194705	0.02327791	0.93283657	0.92266740	1.00000000	0.89064051	0.42582024
8	0.00713355	0.09684970	0.02211401	0.93750075	0.92728074	1.00000000	0.89509371	0.40878743
9	0.00677688	0.09200721	0.02100831	0.94218826	0.93191714	1.00000000	0.89956918	0.39243593
10	0.00643803	0.08740685	0.01995790	0.94689920	0.93657673	1.00000000	0.90406703	0.37673849
11	0.00611613	0.08303651	0.01896000	0.95163369	0.94125961	1.00000000	0.90858736	0.36166895
12	0.00581032	0.07888468	0.01801200	0.95639186	0.94596591	1.00000000	0.91313030	0.34720220
13	0.00551981	0.07494045	0.01711140	0.96117382	0.95069574	1.00000000	0.91769595	0.33331411
14	0.00524382	0.07119343	0.01625583	0.96597969	0.95544922	1.00000000	0.92228443	0.31998154
15	0.00498163	0.06763376	0.01544304	0.97080959	0.96022646	1.00000000	0.92689585	0.30718228
16	0.00473254	0.06425207	0.01467089	0.97566364	0.96502760	1.00000000	0.93153033	0.29489499
17	0.00449592	0.06103946	0.01393734	0.98054195	0.96985273	1.00000000	0.93618798	0.28309919
18	0.00427112	0.05798749	0.01324048	0.98544466	0.97470200	1.00000000	0.94086892	0.27177522
19	0.00405757	0.05508812	0.01257845	0.99037189	0.97957551	1.00000000	0.94557327	0.26090421
20	0.00385469	0.05233371	0.01194953	0.99532375	0.98447339	1.00000000	0.95030113	0.25046805
21	0.00366195	0.04971703	0.01135205	1.00000000	0.98939575	1.00000000	0.95505264	0.24044932
22	0.00347886	0.04723117	0.01078445	1.00000000	0.99434273	1.00000000	0.95982790	0.23083135
23	0.00330491	0.04486962	0.01024523	1.00000000	0.99931445	1.00000000	0.96462704	0.22159810
24	0.00313967	0.04262613	0.00973297	1.00000000	1.00000000	1.00000000	0.96945018	0.21273417
25	0.00298268	0.04049483	0.00924632	1.00000000	1.00000000	1.00000000	0.97429743	0.20422481
26	0.00283355	0.03847009	0.00878400	1.00000000	1.00000000	1.00000000	0.97916892	0.19605581
27	0.00269187	0.03654658	0.00834480	1.00000000	1.00000000	1.00000000	0.98406476	0.18821358
28	0.00255728	0.03471925	0.00792756	1.00000000	1.00000000	1.00000000	0.98898508	0.18068504
29	0.00242941	0.03298329	0.00753118	1.00000000	1.00000000	1.00000000	0.99393001	0.17345764
30	0.00230794	0.03133413	0.00715463	1.00000000	1.00000000	1.00000000	0.99889966	0.16651933
31	0.00219255	0.02976742	0.00679689	1.00000000	1.00000000	1.00000000	1.00000000	0.15985856
32	0.00208292	0.02827905	0.00645705	1.00000000	1.00000000	1.00000000	1.00000000	0.15346422
33	0.00197877	0.02686510	0.00613420	1.00000000	1.00000000	1.00000000	1.00000000	0.14732565
34	0.00187983	0.02552184	0.00582749	1.00000000	1.00000000	1.00000000	1.00000000	0.14143262
35	0.00178584	0.02424575	0.00553611	1.00000000	1.00000000	1.00000000	1.00000000	0.13577532
36	0.00169655	0.02303346	0.00525931	1.00000000	1.00000000	1.00000000	1.00000000	0.13034430
37	0.00161172	0.02188179	0.00499634	1.00000000	1.00000000	1.00000000	1.00000000	0.12513053
38	0.00153114	0.02078770	0.00474652	1.00000000	1.00000000	1.00000000	1.00000000	0.12012531
39	0.00145458	0.01974831	0.00450920	1.00000000	1.00000000	1.00000000	1.00000000	0.11532030
40	0.00138185	0.01876090	0.00428374	1.00000000	1.00000000	1.00000000	1.00000000	0.11070749
41	0.00131276	0.01782285	0.00406955	1.00000000	1.00000000	1.00000000	1.00000000	0.10627919
42	0.00124712	0.01693171	0.00386607	1.00000000	1.00000000	1.00000000	1.00000000	0.10202802
43	0.00118476	0.01608513	0.00367277	1.00000000	1.00000000	1.00000000	1.00000000	0.09794690
44	0.00112553	0.01528087	0.00348913	1.00000000	1.00000000	1.00000000	1.00000000	0.09402902
45	0.00106925	0.01451683	0.00331468	1.00000000	1.00000000	1.00000000	1.00000000	0.09026786
46	0.00101579	0.01379098	0.00314894	1.00000000	1.00000000	1.00000000	1.00000000	0.08665715
47	0.00096500	0.01310144	0.00299149	1.00000000	1.00000000	1.00000000	1.00000000	0.08319086
48	0.00091675	0.01244636	0.00284192	1.00000000	1.00000000	1.00000000	1.00000000	0.07986323
49	0.00087091	0.01182405	0.00269982	1.00000000	1.00000000	1.00000000	1.00000000	0.07666870
50	0.00082737	0.01123284	0.00256483	1.00000000	1.00000000	1.00000000	1.00000000	0.07360195
51	0.00078600	0.01067120	0.00243659	1.00000000	1.00000000	1.00000000	1.00000000	0.07065787
52	0.00074670	0.01013764	0.00231476	1.00000000	1.00000000	1.00000000	1.00000000	0.06783156
53	0.00070936	0.00963076	0.00219902	1.00000000	1.00000000	1.00000000	1.00000000	0.06511829
54	0.00067389	0.00914922	0.00208907	1.00000000	1.00000000	1.00000000	1.00000000	0.06251356
55	0.00064020	0.00869176	0.00198462	1.00000000	1.00000000	1.00000000	1.00000000	0.06001302
56	0.00060819	0.00825717	0.00188539	1.00000000	1.00000000	1.00000000	1.00000000	0.05761250
57	0.00057778	0.00784431	0.00179112	1.00000000	1.00000000	1.00000000	1.00000000	0.05530800
58	0.00054889	0.00745210	0.00170156	1.00000000	1.00000000	1.00000000	1.00000000	0.05309568
59	0.00052145	0.00707949	0.00161648	1.00000000	1.00000000	1.00000000	1.00000000	0.05097185
60	0.00049537	0.00672552	0.00153566	1.00000000	1.00000000	1.00000000	1.00000000	0.04893298
61	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000

	Metric #							
time step	1	2	3	4	5	6	7	8
62	0.20221084	0.11594203	0.12192330	0.13414634	0.05263158	0.25000000	0.14285714	0.02564103
63	0.21232138	0.12173913	0.12801946	0.14085366	0.05526316	0.26250000	0.15000000	0.02692308
64	0.22293745	0.12782609	0.13442044	0.14789634	0.05802632	0.27562500	0.15750000	0.02826923
65	0.23408432	0.13421739	0.14114146	0.15529116	0.06092763	0.28940625	0.16537500	0.02968269
66	0.24578854	0.14092826	0.14819853	0.16305572	0.06397401	0.30387656	0.17364375	0.03116683
67	0.25807796	0.14797467	0.15560846	0.17120850	0.06717271	0.31907039	0.18232594	0.03272517
68	0.27098186	0.15537341	0.16338888	0.17976893	0.07053135	0.33502391	0.19144223	0.03436143
69	0.28453096	0.16314208	0.17155832	0.18875737	0.07405792	0.35177511	0.20101435	0.03607950
70	0.29875750	0.17129918	0.18013624	0.19819524	0.07776081	0.36936386	0.21106506	0.03788347
71	0.31369538	0.17986414	0.18914305	0.20810500	0.08164885	0.38783205	0.22161832	0.03977765
72	0.32938015	0.18885735	0.19860020	0.21851025	0.08573130	0.40722366	0.23269923	0.04176653
73	0.34584916	0.19830022	0.20853021	0.22943577	0.09001786	0.42758484	0.24433419	0.04385486
74	0.36314161	0.20821523	0.21895673	0.24090756	0.09451875	0.44896408	0.25655090	0.04604760
75	0.38129869	0.21862599	0.22990456	0.25295293	0.09924469	0.47141229	0.26937845	0.04834998
76	0.40036363	0.22955729	0.24139979	0.26560058	0.10420693	0.49498290	0.28284737	0.05076748
77	0.42038181	0.24103515	0.25346978	0.27888061	0.10941727	0.51973204	0.29698974	0.05330585
78	0.44140090	0.25308691	0.26614327	0.29282464	0.11488814	0.54571865	0.31183923	0.05597114
79	0.46347095	0.26574125	0.27945043	0.30746587	0.12063254	0.57300458	0.32743119	0.05876970
80	0.48664449	0.27902832	0.29342295	0.32283917	0.12666417	0.60165481	0.34380275	0.06170819
81	0.51097672	0.29297973	0.30809410	0.33898112	0.13299738	0.63173755	0.36099289	0.06479359
82	0.53652555	0.30762872	0.32349881	0.35593018	0.13964725	0.66332443	0.37904253	0.06803327
83	0.56335183	0.32301016	0.33967375	0.37372669	0.14662961	0.69649065	0.39799466	0.07143494
84	0.59151942	0.33916066	0.35665743	0.39241302	0.15396109	0.73131518	0.41789439	0.07500669
85	0.62109539	0.35611870	0.37449031	0.41203367	0.16165915	0.76788094	0.43878911	0.07875702
86	0.65215016	0.37392463	0.39321482	0.43263536	0.16974210	0.80627499	0.46072856	0.08269487
87	0.68475767	0.39262086	0.41287556	0.45426713	0.17822921	0.84658874	0.48376499	0.08682961
88	0.71899556	0.41225191	0.43351934	0.47698048	0.18714067	0.88891817	0.50795324	0.09117109
89	0.75494533	0.43286450	0.45519531	0.50082951	0.19649770	0.93336408	0.53335090	0.09572965
90	0.79269260	0.45450773	0.47795507	0.52587098	0.20632259	0.98003228	0.56001845	0.10051613
91	0.83232723	0.47723311	0.50185283	0.55216453	0.21663872	1.00000000	0.58801937	0.10554194
92	0.87394359	0.50109477	0.52694547	0.57977276	0.22747065	1.00000000	0.61742034	0.11081904
93	0.91764077	0.52614951	0.55329274	0.60876140	0.23884418	1.00000000	0.64829136	0.11635999
94	0.96352281	0.55245698	0.58095738	0.63919947	0.25078639	1.00000000	0.68070592	0.12217799
95	1.00000000	0.58007983	0.61000525	0.67115944	0.26332571	1.00000000	0.71474122	0.12828689
96	1.00000000	0.60908382	0.64050551	0.70471741	0.27649200	1.00000000	0.75047828	0.13470123
97	1.00000000	0.63953801	0.67253078	0.73995328	0.29031660	1.00000000	0.78800220	0.14143629
98	1.00000000	0.67151491	0.70615732	0.77695095	0.30483243	1.00000000	0.82740231	0.14850811
99	1.00000000	0.70509066	0.74146519	0.81579849	0.32007405	1.00000000	0.86877242	0.15593351
100	1.00000000	0.74034519	0.77853845	0.85658842	0.33607775	1.00000000	0.91221104	0.16373019
101	1.00000000	0.77736245	0.81746537	0.89941784	0.35288164	1.00000000	0.95782159	0.17191670
102	1.00000000	0.81623058	0.83517459	0.94438873	0.37052572	1.00000000	1.00000000	0.18051253
103	1.00000000	0.85704210	0.83517459	0.99160817	0.38905201	1.00000000	1.00000000	0.18953816
104	1.00000000	0.89989421	0.83517459	1.00000000	0.40850461	1.00000000	1.00000000	0.19901507
105	1.00000000	0.94488892	0.83517459	1.00000000	0.42892984	1.00000000	1.00000000	0.20896582
106	1.00000000	0.99213337	0.83517459	1.00000000	0.45037633	1.00000000	1.00000000	0.21941411
107	1.00000000	1.00000000	0.83517459	1.00000000	0.47289515	1.00000000	1.00000000	0.23038482
108	1.00000000	1.00000000	0.83517459	1.00000000	0.49653990	1.00000000	1.00000000	0.24190406
109	1.00000000	1.00000000	0.83955111	1.00000000	0.52136690	1.00000000	1.00000000	0.25399926
110	1.00000000	1.00000000	0.84860937	1.00000000	0.54743524	1.00000000	1.00000000	0.26669922
111	1.00000000	1.00000000	0.85812055	1.00000000	0.57480701	1.00000000	1.00000000	0.28003418
112	1.00000000	1.00000000	0.86810729	1.00000000	0.60354736	1.00000000	1.00000000	0.29403589
113	1.00000000	1.00000000	0.87859336	1.00000000	0.63372473	1.00000000	1.00000000	0.30873769
114	1.00000000	1.00000000	0.88960374	1.00000000	0.66541096	1.00000000	1.00000000	0.32417457
115	1.00000000	1.00000000	0.90116464	1.00000000	0.69868151	1.00000000	1.00000000	0.34038330
116	1.00000000	1.00000000	0.91330358	1.00000000	0.73361558	1.00000000	1.00000000	0.35740246
117	1.00000000	1.00000000	0.92604947	1.00000000	0.77029636	1.00000000	1.00000000	0.37527259
118	1.00000000	1.00000000	0.93943265	1.00000000	0.80881118	1.00000000	1.00000000	0.39403622
119	1.00000000	1.00000000	0.95348499	1.00000000	0.84925174	1.00000000	1.00000000	0.41373803
120	1.00000000	1.00000000	0.96823995	1.00000000	0.89171433	1.00000000	1.00000000	0.43442493
121	1.00000000	1.00000000	0.98373266	1.00000000	0.93630004	1.00000000	1.00000000	0.45614618
122	1.00000000	1.00000000	1.00000000	1.00000000	0.98311505	1.00000000	1.00000000	0.47895348

	Metric #							
time step	9	10	11	12	13	14	15	16
62	0.01075269	0.12192330	0.03333333	0.90083060	0.89101034	0.96891192	0.86008230	0.56666667
63	0.01129032	0.12801946	0.03500000	0.85578907	0.84645982	0.92046632	0.81707819	0.53833333
64	0.01185484	0.13442044	0.03675000	0.81299962	0.80413683	0.87444301	0.77622428	0.51141667
65	0.01244758	0.14114146	0.03858750	0.77234964	0.76392999	0.83072085	0.73741307	0.48584583
66	0.01306996	0.14819853	0.04051688	0.73373215	0.72573349	0.78918481	0.70054241	0.46155354
67	0.01372346	0.15560846	0.04254272	0.69704555	0.68944682	0.74972557	0.66551529	0.43847586
68	0.01440963	0.16338888	0.04466985	0.66219327	0.65497448	0.71223929	0.63223953	0.41655207
69	0.01513011	0.17155832	0.04690335	0.62908361	0.62222575	0.67662733	0.60062755	0.39572447
70	0.01588662	0.18013624	0.04924851	0.59762943	0.59111447	0.64279596	0.57059617	0.37593824
71	0.01668095	0.18914305	0.05171094	0.56774795	0.56155874	0.61065616	0.54206636	0.35714133
72	0.01751500	0.19860020	0.05429649	0.53936056	0.53348081	0.58012336	0.51496305	0.33928427
73	0.01839075	0.20853021	0.05701131	0.51239253	0.50680676	0.55111719	0.48921489	0.32232005
74	0.01931028	0.21895673	0.05986188	0.48677290	0.48146643	0.52356133	0.46475415	0.30620405
75	0.02027580	0.22990456	0.06285497	0.46243426	0.45739311	0.49738326	0.44151644	0.29089385
76	0.02128959	0.24139979	0.06599772	0.43931254	0.43452345	0.47251410	0.41944062	0.27634915
77	0.02235407	0.25346978	0.06929761	0.41734692	0.41279728	0.44888839	0.39846859	0.26253170
78	0.02347177	0.26614327	0.07276249	0.39647957	0.39215741	0.42644397	0.37854516	0.24940511
79	0.02464536	0.27945043	0.07640061	0.37665559	0.37254954	0.40512178	0.35961790	0.23693486
80	0.02587763	0.29342295	0.08022064	0.35782281	0.35392207	0.38486569	0.34163701	0.22508811
81	0.02717151	0.30809410	0.08423167	0.33993167	0.33622596	0.36562240	0.32455516	0.21383371
82	0.02853008	0.32349881	0.08844326	0.32293509	0.31941466	0.34734128	0.30832740	0.20314202
83	0.02995659	0.33967375	0.09286542	0.30678833	0.30344393	0.32997422	0.29291103	0.19298492
84	0.03145442	0.35665743	0.09750869	0.29144892	0.28827173	0.31347551	0.27826548	0.18333568
85	0.03302714	0.37449031	0.10238413	0.27687647	0.27385815	0.29780173	0.26435220	0.17416889
86	0.03467849	0.39321482	0.10750333	0.26303265	0.26016524	0.28291165	0.25113459	0.16546045
87	0.03641242	0.41287556	0.11287850	0.24988102	0.24715698	0.26876606	0.23857786	0.15718742
88	0.03823304	0.43351934	0.11852242	0.23738697	0.23479913	0.25532776	0.22664897	0.14932805
89	0.04014469	0.45519531	0.12444854	0.22551762	0.22305917	0.24256137	0.21531652	0.14186165
90	0.04215193	0.47795507	0.13067097	0.21424174	0.21190621	0.23043330	0.20455070	0.13476857
91	0.04425952	0.50185283	0.13720452	0.20352965	0.20131090	0.21891164	0.19432316	0.12803014
92	0.04647250	0.52694547	0.14406475	0.19335317	0.19124536	0.20796606	0.18460700	0.12162863
93	0.04879612	0.55329274	0.15126798	0.18368551	0.18168309	0.19756775	0.17537665	0.11554720
94	0.05123593	0.58095738	0.15883138	0.17450123	0.17259894	0.18768937	0.16660782	0.10976984
95	0.05379773	0.61000525	0.16677295	0.16577617	0.16396899	0.17830490	0.15827743	0.10428135
96	0.05648761	0.64050551	0.17511160	0.15748736	0.15577054	0.16938965	0.15036356	0.09906728
97	0.05931199	0.67253078	0.18386718	0.14961299	0.14798201	0.16092017	0.14284538	0.09411392
98	0.06227759	0.70615732	0.19306054	0.14213234	0.14058291	0.15287416	0.13570311	0.08940822
99	0.06539147	0.74146519	0.20271356	0.13502573	0.13355377	0.14523045	0.12891796	0.08493781
100	0.06866105	0.77853845	0.21284924	0.12827444	0.12687608	0.13796893	0.12247206	0.08069092
101	0.07209410	0.81746537	0.22349171	0.12186072	0.12053227	0.13107048	0.11634845	0.07665637
102	0.07569880	0.83517459	0.23466629	0.11576768	0.11450566	0.12451696	0.11053103	0.07282356
103	0.07948374	0.83517459	0.24639960	0.10997930	0.10878038	0.11829111	0.10500448	0.06918238
104	0.08345793	0.83517459	0.25871959	0.10448033	0.10334136	0.11237656	0.09975426	0.06572326
105	0.08763083	0.83517459	0.27165556	0.09925632	0.09817429	0.10675773	0.09476654	0.06243710
106	0.09201237	0.83517459	0.28523834	0.09429350	0.09326558	0.10141984	0.09002822	0.05931524
107	0.09661299	0.83517459	0.29950026	0.08957883	0.08860230	0.09634885	0.08552681	0.05634948
108	0.10144364	0.83517459	0.31447527	0.08509989	0.08417218	0.09153141	0.08125047	0.05353201
109	0.10651582	0.83955111	0.33019904	0.08084489	0.07996357	0.08695484	0.07718794	0.05085540
110	0.11184161	0.84860937	0.34670899	0.07680265	0.07596539	0.08260710	0.07332854	0.04831263
111	0.11743369	0.85812055	0.36404444	0.07296251	0.07216712	0.07847674	0.06966212	0.04589700
112	0.12330537	0.86810729	0.38224666	0.06931439	0.06855877	0.07455290	0.06617901	0.04360215
113	0.12947064	0.87859336	0.40135899	0.06584867	0.06513083	0.07082526	0.06287006	0.04142205
114	0.13594417	0.88960374	0.42142694	0.06255624	0.06187429	0.06728400	0.05972656	0.03935094
115	0.14274138	0.90116464	0.44249829	0.05942842	0.05878057	0.06391980	0.05674023	0.03738340
116	0.14987845	0.91330358	0.46462320	0.05645700	0.05584155	0.06072381	0.05390322	0.03551423
117	0.15737238	0.92604947	0.48785436	0.05363415	0.05304947	0.05768762	0.05120806	0.03373851
118	0.16524099	0.93943265	0.51224708	0.05095244	0.05039699	0.05480323	0.04864765	0.03205159
119	0.17350304	0.95348499	0.53785944	0.04840482	0.04787715	0.05206307	0.04621527	0.03044901
120	0.18217820	0.96823995	0.56475241	0.04598458	0.04548329	0.04945992	0.04390451	0.02892656
121	0.19128711	0.98373266	0.59299003	0.04368535	0.04320912	0.04698692	0.04170928	0.02748023
122	0.20085146	1.00000000	0.62263953	0.04150108	0.04104867	0.04463758	0.03962382	0.02610622

C.19. Inference on Training Data (Trained on Monotonic Stable-Unstable Data)

Time	First	Second
0	Group 0 1.776618	Group 1 1.772765
1	Group 0 1.787309	Group 1 1.781987
2	Group 0 1.797853	Group 1 1.79114
3	Group 0 1.808234	Group 1 1.800206
4	Group 0 1.818442	Group 1 1.809169
5	Group 0 1.828466	Group 1 1.818015
6	Group 0 1.8383	Group 1 1.826732
7	Group 0 1.847913	Group 1 1.835289
8	Group 0 1.857313	Group 1 1.843689
9	Group 0 1.860668	Group 1 1.846182
10	Group 0 1.85806	Group 1 1.842864
11	Group 0 1.855952	Group 1 1.840101
12	Group 0 1.854296	Group 1 1.837841
13	Group 0 1.865466	Group 1 1.836034
14	Group 0 1.87789	Group 1 1.834637
15	Group 0 1.88994	Group 1 1.83361
16	Group 0 1.901623	Group 1 1.832918
17	Group 0 1.912947	Group 1 1.832527
18	Group 0 1.923807	Group 1 1.832297
19	Group 0 1.932816	Group 1 1.830803
20	Group 0 1.941514	Group 1 1.829554
21	Group 0 1.950948	Group 1 1.828543
22	Group 0 1.962564	Group 1 1.827234
23	Group 0 1.971762	Group 1 1.82448
24	Group 0 1.980319	Group 1 1.821751
25	Group 0 1.98597	Group 1 1.816745
26	Group 0 1.982024	Group 1 1.803159
27	Group 0 1.978164	Group 1 1.790258
28	Group 0 1.974501	Group 1 1.778091
29	Group 0 1.971022	Group 1 1.766609
30	Group 0 1.967715	Group 1 1.755766
31	Group 0 1.964934	Group 1 1.745888
32	Group 0 1.962397	Group 1 1.736662
33	Group 0 1.959989	Group 1 1.727949
34	Group 0 1.957704	Group 1 1.719717
35	Group 0 1.955532	Group 1 1.711935
36	Group 0 1.953469	Group 1 1.704576
37	Group 0 1.951506	Group 1 1.697615
38	Group 0 1.94964	Group 1 1.691028
39	Group 0 1.947864	Group 1 1.684791
40	Group 0 1.946172	Group 1 1.678886
41	Group 0 1.944561	Group 1 1.673291
42	Group 0 1.943027	Group 1 1.66799
43	Group 0 1.941564	Group 1 1.662966
44	Group 0 1.940169	Group 1 1.658202
45	Group 0 1.938838	Group 1 1.653684
46	Group 0 1.937569	Group 1 1.649399
47	Group 0 1.936357	Group 1 1.645332
48	Group 0 1.9352	Group 1 1.641473
49	Group 0 1.934095	Group 1 1.63781
50	Group 0 1.93304	Group 1 1.634331
51	Group 0 1.932032	Group 1 1.631028
52	Group 0 1.931069	Group 1 1.62789
53	Group 0 1.930148	Group 1 1.62491
54	Group 0 1.929267	Group 1 1.622077
55	Group 0 1.928425	Group 1 1.619386
56	Group 0 1.92762	Group 1 1.616827
57	Group 0 1.926849	Group 1 1.614395
58	Group 0 1.926112	Group 1 1.612082
59	Group 0 1.925406	Group 1 1.609884
60	Group 0 1.924731	Group 1 1.607792
61	blank 2	Group 58 0
62	Group 0 1.779737	Group 1 1.775884
63	Group 0 1.781701	Group 1 1.779324
64	Group 0 1.78284	Group 1 1.78202
65	Group 1 1.783087	Group 0 1.782267
66	Group 2 1.783189	Group 1 1.782285
67	Group 3 1.783148	Group 2 1.782152
68	Group 4 1.788709	Group 5 1.787498
69	Group 5 1.93591	Group 4 1.9347
70	Group 6 1.99772	Group 5 1.996386
71	Group 7 1.992383	Group 6 1.990912
72	Group 8 1.9877	Group 7 1.986078
73	Group 9 1.983578	Group 8 1.98179
74	Group 10 1.979942	Group 9 1.977971
75	Group 11 1.976724	Group 10 1.974551
76	Group 12 1.973868	Group 13 1.971227
77	Group 13 1.971326	Group 12 1.968685
78	Group 14 1.969056	Group 13 1.966145
79	Group 15 1.967024	Group 14 1.963815
80	Group 16 1.965199	Group 15 1.961662
81	Group 17 1.963555	Group 16 1.959656
82	Group 18 1.96207	Group 17 1.957772
83	Group 19 1.960725	Group 18 1.955987
84	Group 20 1.959502	Group 19 1.954281
85	Group 21 1.958389	Group 20 1.952634
86	Group 22 1.957373	Group 21 1.95103
87	Group 23 1.956443	Group 22 1.949452
88	Group 24 1.946352	Group 23 1.938648
89	Group 25 1.906624	Group 24 1.898133
90	Group 26 1.888245	Group 27 1.884146
91	Group 27 1.936921	Group 28 1.933771
92	Group 28 1.990091	Group 27 1.986941
93	Group 29 1.987242	Group 30 1.983415
94	Group 30 1.988205	Group 29 1.984377
95	Group 31 1.989188	Group 30 1.984969
96	Group 32 1.990192	Group 33 1.985066
97	Group 33 1.991221	Group 32 1.986095
98	Group 34 1.992277	Group 33 1.986627
99	Group 35 1.993363	Group 34 1.987135
100	Group 36 1.994481	Group 35 1.987618
101	Group 37 1.995635	Group 38 1.988895
102	Group 38 1.996827	Group 39 1.9955
103	Group 39 1.99806	Group 38 1.996733
104	Group 40 1.999338	Group 39 1.997876
105	Group 41 2	Group 40 1.998388
106	Group 42 2	Group 41 1.99751
107	Group 43 2	Group 42 1.99727
108	Group 44 2	Group 43 1.997005
109	Group 45 2	Group 44 1.996651
110	Group 46 2	Group 45 1.996129
111	Group 47 2	Group 46 1.995743
112	Group 48 2	Group 49 1.994845
113	Group 49 2	Group 48 1.994844
114	Group 50 2	Group 51 1.993751
115	Group 51 2	Group 50 1.993751
116	Group 52 2	Group 53 1.992419
117	Group 53 2	Group 52 1.992419
118	Group 54 2	Group 55 1.990612
119	Group 55 2	Group 54 1.990569
120	Group 56 2	Group 57 1.988087
121	Group 57 2	Group 56 1.988088
122	Group 58 2	Group 57 1.986875

C.20. Inference on Actual Data (Trained on Monotonic Stable-Unstable Data)

Time	First	Second	Third
0	Group 4 1.10744	Group 5 1.0764	Group 6 1.043782
1	Group 4 1.29603	Group 5 1.26927	Group 6 1.240051
2	Group 4 1.494941	Group 5 1.48412	Group 6 1.470392
3	Group 4 1.275525	Group 5 1.245897	Group 6 1.214301
4	Group 0 1.968647	Group 1 1.956845	Group 2 1.9429
5	Group 2 1.663143	Group 1 1.662661	Group 3 1.661705
6	Group 0 1.91697	Group 1 1.903157	Group 2 1.887167
7	Group 0 1.579172	Group 1 1.562212	Group 2 1.543141
8	Group 3 1.761756	Group 2 1.760432	Group 1 1.757327
9	Group 0 1.669215	Group 1 1.651918	Group 2 1.632522
10	Group 0 1.682141	Group 1 1.677312	Group 2 1.670526
11	Group 25 1.121081	Group 24 1.118762	Group 23 1.105853
12	Group 22 1.444566	Group 21 1.440598	Group 23 1.435091
13	Group 3 1.668496	Group 2 1.650038	Group 1 1.63119
14	Group 3 1.5965	Group 2 1.575433	Group 1 1.554296
15	Group 0 1.709698	Group 1 1.70499	Group 2 1.698328
16	Group 3 1.441241	Group 2 1.433847	Group 1 1.425541
17	Group 3 1.658548	Group 2 1.648333	Group 1 1.637012
18	Group 12 1.960125	Group 13 1.939044	Group 14 1.911954
19	Group 0 1.574832	Group 1 1.554279	Group 2 1.53177
20	Group 12 1.56223	Group 13 1.561568	Group 14 1.555202
21	Group 0 1.575075	Group 1 1.553756	Group 2 1.530525
22	Group 0 1.67962	Group 1 1.661686	Group 2 1.641676
23	Group 3 1.597114	Group 2 1.58503	Group 1 1.572091
24	Group 12 1.472896	Group 13 1.472407	Group 14 1.466226
25	Group 12 1.34236	Group 13 1.294386	Group 14 1.241357
26	Group 0 1.352998	Group 1 1.346888	Group 2 1.339493
27	Group 12 1.310301	Group 13 1.230233	Group 14 1.148271
28	Group 12 1.180874	Group 0 1.120878	Group 1 1.114366
29	Group 12 1.530219	Group 13 1.515642	Group 14 1.495061
30	Group 12 1.381929	Group 13 1.289375	Group 14 1.196749
31	Group 12 1.397396	Group 0 1.35103	Group 1 1.342926
32	Group 12 1.777904	Group 13 1.690432	Group 14 1.602104
33	Group 12 1.415222	Group 13 1.327305	Group 14 1.238599
34	Group 0 1.344245	Group 1 1.339721	Group 2 1.333971
35	Group 12 1.364022	Group 13 1.280994	Group 14 1.196472
36	Group 12 1.360876	Group 13 1.262856	Group 14 1.165665
37	Group 12 1.2122	Group 13 1.121178	Group 14 1.02984
38	Group 12 1.174856	Group 13 1.082574	Group 14 0.990176
39	Group 12 1.156442	Group 13 1.054621	Group 14 0.954292
40	Group 12 1.195205	Group 13 1.116158	Group 14 1.035081
41	Group 12 1.315752	Group 13 1.218831	Group 14 1.122552
42	Group 12 1.378272	Group 13 1.276451	Group 14 1.176123
43	Group 12 1.639405	Group 13 1.555941	Group 14 1.471043
44	Group 12 1.878105	Group 13 1.80413	Group 14 1.727491
45	Group 12 1.517151	Group 13 1.444452	Group 14 1.368938
46	Group 12 1.330323	Group 13 1.251093	Group 14 1.169857
47	Group 12 1.548336	Group 13 1.463013	Group 14 1.37652
48	Group 12 1.650163	Group 13 1.548342	Group 14 1.448014
49	Group 12 1.44882	Group 13 1.369168	Group 14 1.287562
50	Group 12 1.208305	Group 13 1.141093	Group 14 1.070456
51	Group 12 1.216896	Group 13 1.144381	Group 14 1.069027
52	Group 0 1.68092	Group 1 1.66153	Group 2 1.640124
53	Group 0 1.73642	Group 1 1.718821	Group 2 1.699133
54	Group 0 1.58753	Group 1 1.480808	Group 2 1.454218
55	Group 0 1.715179	Group 1 1.563241	Group 2 1.536651
56	Group 0 1.61419	Group 1 1.543744	Group 2 1.518452
57	Group 0 1.694396	Group 1 1.600579	Group 2 1.575286
58	Group 0 1.818262	Group 1 1.81002	Group 2 1.799717
59	Group 0 1.776493	Group 1 1.771484	Group 2 1.764512

C.21. Inference on $N = 8$ Training Data (Trained on Progressively Stable-Unstable)

Time	First	Second
0	Group 0 1.75188	Group 1 1.014652
1	Group 0 1.757992	Group 1 1.006771
2	Group 0 1.76433	Group 1 0.999294
3	Group 0 1.770856	Group 1 0.992198
4	Group 0 1.777533	Group 1 0.985461
5	Group 0 1.784331	Group 1 0.979064
6	Group 0 1.796844	Group 1 0.972989
7	Group 0 1.814954	Group 1 0.967217
8	Group 0 1.832582	Group 1 0.961734
9	Group 0 1.84973	Group 1 0.956523
10	Group 0 1.866401	Group 1 0.95157
11	Group 0 1.882601	Group 1 0.946861
12	Group 0 1.898336	blank 0.954149
13	Group 0 1.913612	blank 0.963041
14	Group 0 1.927843	blank 0.970979
15	Group 0 1.937377	blank 0.974388
16	Group 0 1.946705	blank 0.977637
17	Group 0 1.955826	blank 0.980731
18	Group 0 1.964278	blank 0.983223
19	Group 0 1.971203	blank 0.984262
20	Group 0 1.978016	blank 0.985249
21	Group 0 1.984714	blank 0.986188
22	Group 0 1.982464	blank 0.987083
23	Group 0 1.977259	blank 0.987933
24	Group 0 1.972332	blank 0.988744
25	Group 0 1.967665	blank 0.989515
26	Group 0 1.963238	blank 0.990248
27	Group 0 1.959034	blank 0.990945
28	Group 0 1.955039	blank 0.991607
29	Group 0 1.951238	blank 0.992237
30	Group 0 1.947618	blank 0.992835
31	Group 0 1.94469	blank 0.993406
32	Group 0 1.942055	blank 0.993949
33	Group 0 1.939554	blank 0.994465
34	Group 0 1.937178	blank 0.994956
35	Group 0 1.934921	blank 0.995422
36	Group 0 1.932775	blank 0.995865
37	Group 0 1.930734	blank 0.996286
38	Group 0 1.928792	blank 0.996687
39	Group 0 1.926942	blank 0.997067
40	Group 0 1.925181	blank 0.997429
41	Group 0 1.923502	blank 0.997772
42	Group 0 1.921902	blank 0.998099
43	Group 0 1.920377	blank 0.998409
44	Group 0 1.918921	blank 0.998704
45	Group 0 1.917532	blank 0.998984
46	Group 0 1.916205	blank 0.99925
47	Group 0 1.914939	blank 0.999504
48	Group 0 1.913729	blank 0.999744
49	Group 0 1.912573	blank 0.999972
50	Group 0 1.911365	blank 1.000085
51	Group 0 1.910102	blank 1.000085
52	Group 0 1.908896	blank 1.000086
53	Group 0 1.907744	blank 1.000086
54	Group 0 1.906643	blank 1.000086
55	Group 0 1.905591	blank 1.000086
56	Group 0 1.904585	blank 1.000086
57	Group 0 1.903623	blank 1.000086
58	Group 0 1.902703	blank 1.000086
59	Group 0 1.901823	blank 1.000086
60	Group 0 1.900981	blank 1.000087
61	blank 2	Group 3 0
62	Group 0 1.730495	Group 1 0.993267
63	Group 0 1.745107	Group 1 1.011967
64	Group 0 1.759768	Group 1 1.036672
65	Group 0 1.774499	Group 1 1.069974
66	Group 0 1.789321	Group 1 1.115615
67	Group 0 1.804259	Group 1 1.178956
68	Group 0 1.81647	Group 1 1.264726
69	Group 0 1.823618	Group 1 1.381204
70	Group 0 1.636247	Group 1 1.419704
71	Group 0 1.471152	Group 1 1.441574
72	Group 1 1.465896	Group 0 1.351508
73	Group 1 1.492541	Group 0 1.264355
74	Group 1 1.521342	Group 0 1.200496
75	Group 1 1.552106	Group 0 1.153401
76	Group 1 1.584622	Group 0 1.118425
77	Group 1 1.618671	Group 0 1.092259
78	Group 1 1.65403	Group 0 1.072535
79	Group 1 1.69048	Group 0 1.057549
80	Group 1 1.727806	Group 0 1.046075
81	Group 1 1.765803	Group 0 1.037218
82	Group 1 1.80428	Group 0 1.030329
83	Group 1 1.779622	Group 0 0.961491
84	Group 1 1.713976	blank 0.896002
85	Group 1 1.651873	blank 0.896927
86	Group 1 1.678562	Group 2 0.964153
87	Group 1 1.724412	Group 2 1.028807
88	Group 1 1.767945	Group 2 1.092466
89	Group 1 1.79138	Group 2 1.140476
90	Group 1 1.792989	Group 2 1.181373
91	Group 1 1.783966	Group 2 1.218984
92	Group 1 1.777393	Group 2 1.25926
93	Group 1 1.773243	Group 2 1.302611
94	Group 1 1.771521	Group 2 1.349501
95	Group 1 1.772266	Group 2 1.400464
96	Group 1 1.77554	Group 2 1.456103
97	Group 1 1.781428	Group 2 1.517109
98	Group 1 1.760823	Group 2 1.561458
99	Group 1 1.752792	Group 2 1.592044
100	Group 1 1.775247	Group 2 1.624306
101	Group 1 1.798449	Group 2 1.658471
102	Group 1 1.815849	Group 2 1.68341
103	Group 1 1.828277	Group 2 1.700207
104	Group 1 1.840781	Group 2 1.717408
105	Group 1 1.853396	Group 2 1.735075
106	Group 1 1.866154	Group 2 1.753273
107	Group 1 1.879091	Group 2 1.772075
108	Group 1 1.892245	Group 2 1.791557
109	Group 1 1.906978	Group 2 1.814189
110	Group 1 1.923467	Group 2 1.840387
111	Group 1 1.940501	Group 2 1.868035
112	Group 1 1.953362	Group 2 1.892803
113	Group 1 1.957339	Group 2 1.910134
114	Group 1 1.961132	Group 2 1.928153
115	Group 1 1.96475	Group 2 1.946936
116	Group 1 1.9682	Group 2 1.966566
117	Group 2 1.971488	Group 1 1.956087
118	Group 2 1.974622	Group 1 1.941644
119	Group 2 1.97761	Group 3 1.951403
120	Group 2 1.980456	Group 3 1.978043
121	Group 3 1.983167	Group 2 1.960497
122	Group 3 1.98575	Group 2 1.937411

C.22. Inference on $N = 4$ Training Data (Trained on Progressively Stable-Unstable)

Time	First	Second
0	Group 0 1.939798	Group 2 1.491293
1	Group 0 1.94435	Group 2 1.516556
2	Group 0 1.948706	Group 2 1.541246
3	Group 0 1.952873	Group 2 1.565349
4	Group 0 1.956858	Group 2 1.588854
5	Group 0 1.960667	Group 2 1.611754
6	Group 0 1.964308	Group 2 1.634044
7	Group 0 1.967786	Group 2 1.655723
8	Group 0 1.971108	Group 2 1.67679
9	Group 0 1.97428	Group 2 1.697248
10	Group 0 1.977308	Group 2 1.7171
11	Group 0 1.980197	Group 2 1.736354
12	Group 0 1.982955	Group 2 1.755014
13	Group 0 1.985585	Group 2 1.77309
14	Group 0 1.988094	Group 2 1.790591
15	Group 0 1.990486	Group 2 1.807526
16	Group 0 1.992766	Group 2 1.823907
17	Group 0 1.99494	Group 2 1.839744
18	Group 0 1.997012	Group 2 1.85505

Time	First	Second
19	Group 0 1.998986	Group 2 1.869836
20	Group 0 1.999409	Group 2 1.882658
21	Group 0 1.998193	Group 2 1.893435
22	Group 0 1.997041	Group 2 1.90388
23	Group 0 1.99595	Group 2 1.914
24	Group 0 1.994917	Group 2 1.923799
25	Group 0 1.993938	Group 2 1.933284
26	Group 0 1.993011	Group 2 1.942461
27	Group 0 1.992132	Group 2 1.951339
28	Group 0 1.991299	Group 2 1.959922
29	Group 0 1.99051	Group 2 1.968218
30	Group 0 1.989762	Group 2 1.976235
31	Group 0 1.989052	Group 2 1.983979
32	Group 2 1.988379	Group 0 1.98531
33	Group 2 1.987742	Group 0 1.976922
34	Group 2 1.987137	Group 0 1.96896
35	Group 2 1.986563	Group 0 1.961399
36	Group 2 1.986019	Group 0 1.954215
37	Group 2 1.985503	Group 0 1.947388

Time	First	Second
38	Group 2 1.985013	Group 0 1.940897
39	Group 2 1.984549	Group 0 1.934722
40	Group 2 1.984108	Group 0 1.928847
41	Group 2 1.98369	Group 0 1.923255
42	Group 2 1.983293	Group 0 1.91793
43	Group 2 1.982916	Group 0 1.912859
44	Group 2 1.982559	Group 0 1.908027
45	Group 2 1.98222	Group 0 1.903423
46	Group 2 1.981898	Group 0 1.899033
47	Group 2 1.981592	Group 0 1.894848
48	Group 2 1.981302	Group 0 1.890856
49	Group 2 1.981027	Group 0 1.887048
50	Group 2 1.980766	Group 0 1.883414
51	Group 2 1.980518	Group 0 1.879946
52	Group 2 1.980282	Group 0 1.876635
53	Group 2 1.980059	Group 0 1.873474
54	Group 2 1.979846	Group 0 1.870455
55	Group 2 1.979645	Group 0 1.867572
56	Group 2 1.979454	Group 0 1.864816

Time	First	Second
57	Group 2 1.979272	Group 0 1.862184
58	Group 2 1.9791	Group 0 1.859668
59	Group 2 1.978936	Group 0 1.857264
60	Group 2 1.97878	Group 0 1.854965
61	blank 2	Group 2 0
62	Group 0 1.939798	Group 2 1.491293
63	Group 0 1.93528	Group 2 1.51274
64	Group 0 1.930573	Group 2 1.533392
65	Group 0 1.92567	Group 2 1.553206
66	Group 0 1.920566	Group 2 1.572144
67	Group 0 1.915254	Group 2 1.590179
68	Group 0 1.909729	Group 2 1.607289
69	Group 0 1.903986	Group 2 1.623457
70	Group 0 1.77344	Group 2 1.529057
71	Group 0 1.580189	Group 2 1.378897
72	Group 0 1.414007	Group 2 1.248151
73	Group 0 1.270605	Group 1 1.222715
74	Group 1 1.277449	Group 0 1.14643
75	Group 1 1.335893	Group 0 1.038527

Time	First	Second
76	Group 1 1.398609	blank 1.048423
77	Group 1 1.46631	blank 1.096347
78	Group 1 1.539887	blank 1.142522
79	Group 1 1.620462	blank 1.186813
80	Group 1 1.707318	blank 1.227838
81	Group 1 1.728171	blank 1.224677
82	Group 1 1.748395	blank 1.219373
83	Group 1 1.76804	blank 1.212504
84	Group 1 1.787166	blank 1.204651
85	Group 1 1.805839	blank 1.196385
86	Group 1 1.824127	blank 1.188253
87	Group 1 1.842099	blank 1.180761
88	Group 1 1.851774	blank 1.166502
89	Group 1 1.845072	blank 1.148982
90	Group 1 1.83924	blank 1.134479
91	Group 1 1.843092	blank 1.123069
92	Group 1 1.840302	blank 1.105988
93	Group 1 1.83095	blank 1.084632
94	Group 1 1.822755	blank 1.066581

Time	First	Second
95	Group 1 1.816738	blank 1.054918
96	Group 1 1.814744	blank 1.055049
97	Group 1 1.812881	blank 1.055174
98	Group 1 1.811139	blank 1.055293
99	Group 1 1.80951	blank 1.055406
100	Group 1 1.807984	blank 1.055514
101	Group 1 1.806554	blank 1.055616
102	Group 1 1.805212	blank 1.055713
103	Group 1 1.803954	blank 1.055806
104	Group 1 1.802771	blank 1.055894
105	Group 1 1.80166	blank 1.055978
106	Group 1 1.800615	blank 1.056058
107	Group 1 1.799632	blank 1.056134
108	Group 1 1.798707	blank 1.056207
109	Group 1 1.797836	blank 1.056275
110	Group 1 1.797015	blank 1.056341
111	Group 1 1.796241	blank 1.056403
112	Group 1 1.795511	blank 1.056462
113	Group 1 1.794823	blank 1.056518
114	Group 1 1.794174	blank 1.056572
115	Group 1 1.79356	blank 1.056623
116	Group 1 1.792981	blank 1.056671
117	Group 1 1.792434	blank 1.056717
118	Group 1 1.791917	blank 1.056761
119	Group 1 1.791429	blank 1.056803
120	Group 1 1.790967	blank 1.056842
121	Group 1 1.79053	blank 1.05688
122	Group 1 1.790117	blank 1.056915

C.23. Inference on Real Data (Permutation #3)

Time	First	Second	Third
0	blank 1.888611	Group 22 1.304987	Group 21 1.304611
1	Group 6 1.543825	Group 7 1.492054	Group 8 1.435806
2	Group 11 1.956717	Group 12 1.954608	Group 10 1.953776
3	Group 0 1.814317	Group 1 1.76647	Group 2 1.715792
4	Group 5 1.84994	Group 4 1.82054	Group 0 1.815253
5	Group 0 1.872662	Group 1 1.827132	Group 2 1.792969
6	Group 5 1.718444	Group 4 1.681672	Group 3 1.601874
7	Group 0 1.942873	Group 1 1.91218	Group 2 1.889489
8	Group 2 1.959733	Group 1 1.945374	Group 0 1.942192
9	Group 0 1.910353	Group 1 1.869118	Group 2 1.829161
10	Group 0 1.900571	Group 1 1.859295	Group 2 1.829206
11	Group 27 1.459493	Group 26 1.430269	Group 25 1.403812
12	Group 6 1.595294	Group 7 1.553854	Group 8 1.548791
13	Group 0 1.924894	Group 1 1.897435	Group 2 1.885466
14	Group 0 1.87139	Group 1 1.825816	Group 2 1.795536
15	Group 0 1.806854	Group 1 1.759721	Group 2 1.738661
16	Group 0 1.902507	Group 2 1.89289	Group 1 1.878425
17	Group 2 1.899203	Group 4 1.893445	Group 3 1.879465
18	Group 27 1.563725	Group 26 1.537029	Group 25 1.512371
19	Group 4 1.914185	Group 5 1.901325	Group 3 1.87347
20	Group 24 1.339709	Group 25 1.333547	Group 26 1.327182
21	Group 2 1.912854	Group 4 1.909916	Group 3 1.89773
22	Group 2 1.927765	Group 1 1.915502	Group 3 1.915231
23	Group 4 1.915673	Group 2 1.900416	Group 3 1.894714
24	Group 4 1.849141	Group 5 1.822595	Group 2 1.812618
25	Group 4 1.887702	Group 2 1.8826	Group 3 1.852871
26	Group 0 1.881177	Group 1 1.847927	Group 2 1.844596
27	Group 4 1.807161	Group 5 1.780345	Group 2 1.776865
28	Group 2 1.893184	Group 4 1.886514	Group 0 1.864721
29	Group 24 1.565573	Group 25 1.542482	Group 26 1.519272
30	Group 4 1.702649	Group 5 1.682728	Group 2 1.660002
31	Group 2 1.907993	Group 0 1.894724	Group 4 1.887795
32	Group 4 1.851736	Group 5 1.834333	Group 2 1.805062
33	Group 4 1.8707	Group 2 1.862123	Group 3 1.840595
34	Group 0 1.857416	Group 2 1.844991	Group 1 1.836947
35	Group 4 1.865759	Group 2 1.85002	Group 5 1.825558
36	Group 4 1.759443	Group 2 1.731234	Group 5 1.722252
37	Group 0 1.791727	Group 2 1.78566	Group 4 1.755213
38	Group 0 1.750901	Group 2 1.723955	Group 1 1.710773
39	Group 4 1.840101	Group 2 1.837216	Group 0 1.805058
40	Group 4 1.474998	Group 5 1.457192	Group 2 1.433978
41	Group 24 1.268243	Group 25 1.247856	Group 26 1.227564
42	Group 4 1.38154	Group 5 1.363643	Group 2 1.344887
43	Group 24 1.030091	Group 25 1.022371	Group 26 1.014709
44	Group 24 1.976667	Group 25 1.93803	Group 26 1.899243
45	Group 4 1.351831	Group 5 1.336143	Group 2 1.320466
46	Group 4 1.42394	Group 5 1.420679	Group 3 1.372189
47	Group 24 1.357161	Group 25 1.33299	Group 26 1.308976
48	Group 27 1.256268	Group 26 1.255369	Group 25 1.254531
49	Group 2 1.794651	Group 4 1.793595	Group 0 1.774117
50	Group 4 1.8498	Group 2 1.847169	Group 0 1.816191
51	Group 0 1.792218	Group 2 1.769243	Group 1 1.752257
52	Group 0 1.871231	Group 2 1.833293	Group 1 1.831593
53	Group 0 1.906105	Group 1 1.862149	Group 2 1.847268
54	Group 0 1.939847	Group 1 1.904117	Group 2 1.883682
55	Group 0 1.971246	Group 1 1.94743	Group 2 1.93058
56	Group 0 1.968207	Group 2 1.95222	Group 1 1.950662
57	Group 2 1.967628	Group 0 1.962522	Group 1 1.96143
58	Group 0 1.964331	Group 2 1.963718	Group 1 1.952889
59	Group 0 1.96489	Group 1 1.931914	Group 2 1.888646

APPENDIX D

A NOTE ON VERSIONS

Throughout this work, we have created, trained and tested networks on the Numenta Platform for Intelligent Computing (NuPIC). All calculations have been done with NuPIC (version 1.6.1). This platform is built on Python (version 2.5.2). To ease execution and presentation of results, the Vitamin D Toolkit (version 1.3.0) has been used to run NuPIC. During the course of this research, all of these versions of Python, NuPIC and Vitamin D Toolkit were freely available on the internet:

- Python 2.5.2 (http://www.python.org/download/releases/2.5.2/)
- NuPIC 1.6.1 (http://www.numenta.com/)
- Vitamin D Toolkit 1.3.0 (http://www.vitamindinc.com/toolkit.html)

It should be noted that slightly different results are obtained if networks are created, trained and tested directly in NuPIC, as opposed to using the Vitamin D interface. For instance, the values in Markov-chain probabilities are slightly different. It is for this reason that the significant figures beyond experimental accuracy have been included throughout this work. Nevertheless, for all examples surveyed in this regard, the trends are identical. It is believed that the slight differences in values (e.g., probability values) are due to round-off error between the NuPIC and Vitamin D Toolkit. This possibility has been confirmed by members of the Numenta team in extensive discussion on the issue. Consequently, if anyone attempts to reproduce the results demonstrated in this work then he/she should be able to do so using all of the program versions listed above.

APPENDIX E

GLOSSARY

Although definitions of useful terms have been given when needed, this appendix lists each of those terms that are instrumental to this research. This list is not meant to be exhaustive but it is intended to provide a useful reference. Since three disciplines are combined in this work, the list follows these three divisions. In particular, we present below terms pertaining to complex systems, situational awareness and machine learning, specifically focusing on Hierarchical Temporal Memory terminology.

Complex Systems Terminology

Schema

A schema is a compression of information used for predicting and explaining a complex system's behavior. A possible example of a schema is a neural network trained on data describing complex system.

Coarse Graining

The coarse graining is a specification of the level of detail up to which a complex system is described. This specification can also indicate what finer details are being ignored.

Context

The context specifies what mechanism is used to comprehend a complex system. Some possible mechanisms are other humans, computer-based systems (e.g., HTM networks), etc. The context specifies the coarse graining of a complex system's analysis. Consequently, in this work, we have focused on two contexts (e.g., the shockwaves and

Iraq contexts). For both, the level of detail has been set by the data at hand and our mechanisms for analysis (human- and HTM-based SA).

Ergodicity

Ergodicity is an averaging over fast microscopic dynamics on the time scale of macroscopic observations, causing an averaging over microscopic spatial variations. It is the root of 'smoothness' assumptions in the analysis of physical systems. For complex systems though, time dependence is slow on a microscopic scale, so they do not demonstrate ergodicity.

Situational Awareness Terminology

Goals

Goals are what focus the attention of one's SA onto relevant information. As goals shift, the relevance of a given dataset can change.

Level 1 SA

The first level of situational awareness is to perceive the elements of the environment. The relevance of certain elements over others is determined via the goals.

Level 2 SA

The second level of situational awareness is to comprehend the current situation. This step involves an integration of the perceived elements into a changing schema, whose evolution is dictated via goals as well.

Level 3 SA

The third level of situational awareness is to project the current situation into the future. This projection is built on the previous two levels of SA.

Hierarchical Temporal Memory Terminology

Hierarchical Temporal Memory

Hierarchical Temporal Memory (HTM) is a memory system that exploits the hierarchical structure of its world. It observes this world via data fed into its sensors.

Machine Learning

Machine learning is the study of computer algorithms that improve automatically through experience given by data.

Unsupervised Learning

Unsupervised learning is a process by which algorithms learn directly from data without external guidance.

Supervised Learning

Supervised learning is a process by which algorithms learn both from data and external guidance.

Vector

A vector is an object that exists in a vector space. In this research, we have focused on finite-dimensional vector spaces. These mathematical objects have served as a background on which to describe the learning problem executed with HTM networks.

Evidence

Evidence is the term used to describe data presented to a network. In this research, all evidence has been in the form of finite-dimensional vectors.

Thread

A thread is the informal term given to a set of linked spatial patterns. It is the forerunner of the Markov-chain in HTM theory.

Node

A node is an object defined in the Numenta Platform for Intelligent Computing. There are many types of nodes, including sensor nodes, effector nodes and algorithmic nodes (e.g., the Zeta1 Node).

Sensor Node

A sensor node is the portal through which information enters an HTM network.

Effector Node

An effector node is the portal through which information exits an HTM network.

Network

A network is an object defined in the Numenta Platform for Intelligent Computing. It is made of nodes.

Receptive Field

The receptive field is the effective input area of a node.

Hierarchy

A hierarchy refers to the arrangement of nodes in a network.

Bottom-level

The bottom-level of a network refers to those nodes that are closest to the inputs of the network's receptive field.

Top-level

The top-level of a network refers to those nodes that are farthest from the inputs of the network's receptive field.

Parent and Child Nodes

Given two nodes connected vertically, the one above is the parent and the one below is the child.

Coincidence Pattern

A coincidence pattern – simply, coincidence – is a vector determined to be distinct from other patterns in a node's memory. Each coincidence pattern is stored in a coincidence matrix (C).

Coincidence Matrix

A coincidence matrix (C) contains all of the coincidence patterns of a given node. For example, the coincidence matrix of the first node of the second level of a network would be denoted as $C^{2,1}$. Note that in NuPIC the nodes are counted from zero, but in the text we start the counting from one.

Spatial Pooling

The formation of a coincidence matrix is also called spatial pooling.

Transition Probability

The transition probability between the i^{th} and j^{th} coincidence pattern is the probability that the latter will follow the former in consecutive time steps. Transition probabilities are crucial in forming Markov-chains.

Markov-chain

A Markov-chain is a set of coincidence patterns that are likely to follow one another in consecutive time steps. The set of all Markov-chains of a given node is labeled *G*. For instance, the set of all Markov-chains of the second node of the third-level would be $G^{3,2}$.

Temporal grouping

Temporal grouping is the process of forming *G* for each node.

Time Point (or Time Step)

A time point – or time step – refers to the amount of time between successive pieces of evidence. For example, a time point in the Iraq context lasted one month.

REFERENCES

[1] GELL-MANN, M., *The Quark and the Jaguar*. Henry Holt and Company, 1994.

[2] BAR YAM, Y., *Dynamics of Complex Systems*. Westview Press, 1997.

[3] BOYCE, W. E., and DIPRIMA, R. C., *Elementary Differential Equations and Boundary Value Problems*. Wiley, 2005.

[4] TENENBAUM, M., and POLLARD, H., *Ordinary Differential Equations: An Elementary Textbook for Students of Mathematics, Engineering, and the Sciences*. Dover, 1985.

[5] CODDINTON, E., *An Introduction to Ordinary Differential Equations*. Dover, 1989.

[6] WEISBUCH, G., *Complex System Dynamics*. Westview Press, 1991.

[7] ENDSLEY, M., "Toward a theory of situation awareness in dynamic systems," Human Factors, vol. 37, no. 1, pp. 32–64, 1995.

[8] ALBERTS, D. S., GARSTKA, J. J., and STEIN, F. P., *Network Centric Warfare: Developing and Leveraging Information Superiority*. CCRP Publication Series, 2000.

[9] KAEMPF, G. L., WOLF, S., and MILLER, T. E., "Decision-making in the AEGIS combat information center," pp. 1107-1111, Presented in the Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting, Santa Monica, CA, 1993.

[10] KLEIN, G. A., "Recognition-primed decisions," *Advanced in Man-Machine Systems Research*. W. B. Rouse (Ed.), JAI Press, 1988.

[11] KLEIN, G. A., CALDERWOOD, R., and CLINTON-CIROCCO, A., "Rapid decision making on the fire ground," pp. 576-580, Presented in the Proceedings of the Human Factors Society 30th Annual Meeting, Santa Monica, CA, 1986.

[12] NOBLE, D., BOEHM-DAVIS, D., and GROSZ, C., *Rules, Schema and Decision Making*. Engineering Research Associates Inc., 1987.

[13] SWELLER, J., "Cognitive load during problem solving: Effects on learning," Cognitive Science, vol. 12, pp. 257–285, 1988.

[14] FEDERICO, P., "Expert and novice recognition of similar situations," Human Factors, vol. 37, no. 1, pp. 105–122, 1995.

[15] DREYFUS, H., and DREYFUS, S., *Mind Over Machine: The Power of Human Intuition and Expertise in the Era of the Computer*. Free Press, 1986.

[16] DE GROOT, A. D., *Thought and Choice in Chess*. Mouton Publishers, 1965.

[17] MINTZBERG, H., *The Nature of Managerial Work*. Harper & Row, 1973.

[18] KUHN, T. S., *The Structure of Scientific Revolutions*. University of Chicago Press, 1962.

[19] WICKENS, C. D., *Engineering Psychology and Human Performance*. Harper Collins, 2nd Ed., 1992.

[20] PHISTER JR., P. W., and PLONISCH, I. G., *Information and Knowledge Centric Warfare: The Next Steps in the Evolution of Warfare*. Air Force Research Lab (Rome, NY) Information Directorate, 2004.

[21] MUELLNER, G., "Interoperability of a myriad of emerging broadband capabilities will become key," Aviation Week & Space Technology, December 15, 2003.

[22] BUSCH, T. E., "The theoretical underpinnings of predictive battlespace awareness (PBA)," Air Force Research Laboratory Technical Report, AFRL-IF-RS-TR-2003-187, July 2003.

[23] HALL, D. L., and LLINAS, J., "An introduction to multisensor data fusion," Proceedings of the IEEE, vol. 85, no. 1, pp. 6–23, 1997.

[24] LLINAS, J., and WALTZ, E., *Multisensor Data Fusion*. Artech House, 1990.

[25] HALL, D., *Mathematical Techniques in Multisensor Data Fusion*. Artech House, 1992.

[26] KLEIN, L. A., *Sensor and Data Fusion Concepts and Applications*. SPIE Opt. Engineering Press, Tutorial Texts, vol. 14, 1993.

[27] HALL, D. L., and LLINAS, J., "A challenge for the data fusion community I: Research imperatives for improved processing," Presented at the Proceedings of the 7th National Symposium on Sensor Fusion, Albuquerque, NM, March 1994.

[28] LLINAS, J., and HALL, D. L., "A challenge for the data fusion community II: Infrastructure imperatives," Presented at the Proceedings of the 7th National Symposium on Sensor Fusion, Albuquerque, NM, March 1994.

[29] HALL, D. L., and LINN, R. J., "Survey of commercial software for multisensor data fusion," Presented at the Proceedings of the SPIE Conference on Sensor Fusion and Aerospace Applications, Orlando, FL, April 1991.

[30] KESSLER, O., *Functional Description of the Data Fusion Process*. Technical Report, Office of Naval Technology, Naval Air Development Center, Warminster, PA, January 1992.

[31] WRIGHT, F., "The fusion of multi-source data," Signal, pp. 39–43, October 1980.

[32] SALERNO, J., HINMAN, D., BOULWARE, D., and BELLO, P., "Information fusion for situational awareness," Proceedings of the 6th International Conference on Information Fusion, 2003.

[33] SALERNO, J., HINMAN, D., and BOULWARE, D., "Building a framework for situation awareness," Proceedings of the 7th International Conference on Information Fusion, 2004.

[34] MITCHELL, T., *Machine Learning*. McGraw Hill, 1997.

[35] WITTEN, I., FRANK, E., and GRAY, J., *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufman Publishers, 1999.

[36] DZEROSKI, S., "Multi-relational data mining: An introduction," Newsletter of the ACM Special Interest Group on Knowledge Discovery and Data Mining, vol. 5, no. 1, pp. 1–16, 2003.

[37] JENSEN, D., “Statistical challenges to inductive inference in linked data,” Preliminary Papers of the Seventh International Workshop on Artificial Intelligence and Statistics, 1999.

[38] BRANNON, N. G., SEIFFERTT, J. E., DRAELOS, T. J., and WUNSCH, D. C., “Coordinated machine learning and decision support for situation awareness,” Neural Networks, vol. 22, no. 3, pp. 316–325, 2009.

[39] PAUL, J. L., “Smart sensor web: Web-based exploitation of sensor fusion for visualization of the tactical battlefield,” Aerospace and Electronic Systems Magazine, IEEE, vol. 16, no. 5, pp. 29–36, 2002.

[40] CARPENTER, G., and GROSSBERG, S., “The ART of adaptive pattern recognition by a self-organizing neural network,” Computer, vol. 21, no. 3, pp. 77–87, 1988.

[41] AMIS, G., and CARPENTER, G., “Default ARTMAP 2,” Proceedings of the international joint conference on neural networks, 2007.

[42] BRANNON, N., CONRAD, G., DRAELOS, T., SEIFFERT, J., and WUNSCH, D., “Information fusion and situation awareness using ARTMAP and partially observable Markov decision processes,” Proceedings of the international joint conference on neural networks, 2006.

[43] CARPENTER, G., GROSSBERG, S., MARKOZON, N., REYNOLDS, J., and ROSEN, D., “Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps,” IEEE Transactions on Neural Networks, vol. 23, no. 5, pp. 698–713, 1992.

[44] SAHAMI, M., DUMAIS, S., HECKERMAN, D., and HORVITZ, E., “A Bayesian approach to filtering junk e-mail,” AIAA Technical Report WS-98-05, Presented at the AIAA Workshop on Learning for Text Categorization, July 1998.

[45] KOHL, N., and MIIKKULAINEN, R., “Evolving neural networks for strategic decision-making problems,” Neural Networks, vol. 22, pp. 326–337, 2009.

[46] JORDAN, C., “Sur la série de Fourier,” C. R. Acad. Sci. Paris Sér. I Math., vol. 92, no. 5, pp. 228–230, 1881.

[47] GOLUBOV, B. I., "Variation of a function", http://eom.springer.de/V/v096110.htm (Accessed February 21, 2010).

[48] GUTMANN, H., "A radial basis function method for global optimization," Journal of Global Optimization, vol. 19, no. 3, pp. 201–227, 2001.

[49] MOODY, J., and DARKEN, C. J., "Fast learning in networks of locally tuned processing units," Neural Computation, vol. 1, no. 2, pp. 281–294, 1989.

[50] PARK, J., and SANDBERG, I. W., "Universal approximation using radial-basis function networks," Neural Computation, vol. 3, no. 2, pp. 246–257, 1991.

[51] TULAI, A. F., and OPPACHER, F., "Combining competitive and cooperative coevolution for training cascade neural networks," Proceedings of the genetic and evolutionary computation conference, pp. 618-625, 2002.

[52] STONE, P., KUHLMANN, G., TAYLOR, M. E., and LIU, Y., "Keepaway soccer: From machine learning testbed to benchmark," *Robocup-2005: Robot soccer world cup IX: Vol. 4020*. I. Noda, A. Jacoff, A. Bredenfeld, and Y. Takahasi (Eds.), Springer Verlag, 2006.

[53] POTTER, M. A., and JONG, K. A. D., "Cooperative coevolution: An architecture for evolving coadapted subcomponents," Evolutionary Computation, vol. 8, no. 1, pp. 1–29, 2000.

[54] GEORGE, D., and HAWKINS, J., "Towards a mathematical theory of cortical micro-circuits," PLoS Computational Biology, vol. 5, Issue 10, pp. 1–26, 2009.

[55] GEORGE, D., How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition. Ph.D. Dissertation, Stanford University, 2008.

[56] HAWKINS, J., and BLAKESLEE, S., *On Intelligence*. Owl Books, 2004.

[57] SIMON, H. A., *The Sciences of the Artificial*. MIT Press, 1981.

[58] SCHEY, N. C., Song Identification Using the Numenta Platform for Intelligent Computing. B.S. Thesis, Ohio State University, 2008.

[59] BONHOFF, G. M., Using Hierarchical Temporal Memory for Detecting Anomalous Network Activity. Ph.D. Dissertation, Air Force Institute of Technology, 2008.

[60] PEREA, A. J., MERONO, J. E., and AGUILERA, M. J., "Application of Numenta Hierarchical Temporal Memory for land-use classification," South African Journal of Science, vol. 105, no. 9-10, pp. 370–375, 2009.

[61] PATTEE, H. H., *Hierarchy Theory: The Challenge of Complex Systems*. G. Brazilier, 1973.

[62] GERMAN, S., POTTER, D. F., and CHI, Z., "Composition systems," Quarterly of Applied Mathematics, vol. 60, no. 4, pp. 707–736, 2002.

[63] CHANGIZI, M. A., "Universal scaling laws for hierarchical complexity in languages, organisms, behaviors and other combinatorial systems," Journal of Theoretical Biology, vol. 211, no. 3, pp. 277–295, 2001.

[64] CHECKLAND, P., *Systems Thinking, Systems Practice*. Wiley, 1981.

[65] FUKUSHIMA, K., MIYAKE, S., and ITO, T., "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *Artificial Neural Networks: Theoretical Concepts*. IEEE Computer Society Press, pp. 136-144, 1988.

[66] LECUN, Y., and BENGIO, Y., *Convolutional networks for images, speech, and time-series*. 1995.

[67] RIESENHUBER, M., and POGGIO, T., "Hierarchical models of object recognition in cortex," Nature Neuroscience, vol. 2, no. 11, pp. 1019–1025, 1999.

[68] ULLMAN, S., "Object recognition and segmentation by a fragment-based hierarchy," Trends in Cognitive Sciences, vol. 11, no. 2, pp. 58–64, 2007.

[69] FINE, S., SINGER, Y., and TISHBY, N., "The hierarchical hidden Markov model: analysis and applications," Machine Learning, vol. 32, no. 1, pp. 41–62, 1998.

[70] DOUGLAS, R. J., and MARTIN, K. A., "Neuronal circuits of the neocortex," Annual Review of Neuroscience, vol. 27, pp. 419–451, 2004.

[71] FELLEMAN, D. J., and VAN ESSEN, D. C., "Distributed hierarchical processing in the primate cerebral cortex," Cerebral Cortex, vol. 1, no. 1, pp. 1–47, 1991.

[72] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", July 2005, http://www.defense.gov/news/Jul2005/d20050721secstab.pdf (Accessed February 22, 2010).

[73] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", October 2005, http://www.defense.gov/pubs/20051013_publication_OSSRF.pdf (Accessed February 22, 2010).

[74] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", February 2006, http://www.defense.gov/home/features/Iraq_Reports/docs/2006-02-Report.pdf (Accessed February 22, 2010).

[75] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", May 2006, http://www.defense.gov/pubs/pdfs/May%2006%20Security%20and%20Stabilty%20Report%20Final%20with%20errata.pdf (Accessed February 22, 2010).

[76] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", August 2006, http://www.defense.gov/pubs/pdfs/Security-Stabilty-ReportAug29r1.pdf (Accessed February 22, 2010).

[77] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", November 2006, http://www.defense.gov/pubs/pdfs/9010Quarterly-Report-20061216.pdf (Accessed February 22, 2010).

[78] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", March 2007, http://www.defense.gov/pubs/pdfs/9010_March_2007_Final_Signed.pdf (Accessed February 22, 2010).

[79] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", June 2007,

http://www.defense.gov/pubs/pdfs/9010-Final-20070608.pdf (Accessed February 22, 2010).

[80] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", September 2007, http://www.defense.gov/pubs/pdfs/Signed-Version-070912.pdf (Accessed February 22, 2010).

[81] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", December 2007, http://www.defense.gov/pubs/pdfs/FINAL-SecDef%20Signed-20071214.pdf (Accessed February 22, 2010).

[82] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", March 2008, http://www.defense.gov/pubs/pdfs/Master%20%20Mar08%20-%20final%20signed.pdf (Accessed February 22, 2010).

[83] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", June 2008, http://www.defense.gov/pubs/pdfs/Master_16_June_08_%20FINAL_SIGNED%20.pdf (Accessed February 22, 2010).

[84] O'HANLON, M. E., and CAMPBELL, J., *Iraq Index: Tracking Variables of Reconstruction and Security in Post-Saddam-Iraq*. The Brookings Institution, 2008.

[85] Anon., "Security Council extends mandate of multinational force in Iraq until 31 December 2007, unanimously adopting resolution 1723", http://www.un.org/News/Press/docs/2006/sc8879.doc.htm (Accessed January 21, 2010).

[86] O'HANLON, M. E., and POLLACK, K. M., "Stability in Iraq: A war we just might win", http://www.nytimes.com/2007/07/30/opinion/30pollack.html (Accessed February 22, 2010).

[87] CLARK, W., "Next move in Iraq?", http://blogs.usatoday.com/oped/2006/11/illustration_by_2.html (Accessed February 22, 2010).

[88] Anon., “Americans doubtful about eventual Iraq stability”, http://www.angus-reid.com/polls/view/9616 (Accessed February 22, 2010).

[89] HO, Y. C., and PEPYNE, D. L., “Simple explanation of the no-free-lunch-theorem and its implications,” Journal of Optimization Theory, vol. 115, Issue 3, pp. 549–570, 2009.

[90] WEISS, S. M., and INDURKHYA, N., *Predictive Data Mining: A Practical Guide*. Morgan Kaufmann, 1998.

[91] BLUM, A., and CHAWLA, S., “Learning from labeled and unlabeled data using graph mincuts,” Proceedings of the 18th International Conference on Machine Learning, 2001.

[92] BLUM, A., LAFFERT, J., RWEBANGIRA, M., and REDDY, R., “Semi-supervised learning using randomized mincuts,” Presented at the 21st International Conference on Machine Learning, 2004.

[93] BREFELD, U., and SCHEFFER, T., “Semi-supervised learning for structured output variables,” Presented at the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006.

[94] BURGES, C. J., and PLATT, J. C., “Semi-supervised learning with conditional harmonic mixing,” *Semi-supervised learning*. O. Chapelle, B. Schölkopf and A. Zien (Eds.), MIT Press, 2005.

[95] CRISTIANINI, N., and SHAWE-TAYLOR, J., *An Introduction to Support Vector Machines: And Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

[96] RIEDMILLER, M., “Advanced supervised learning in multi-layer perceptrons – From backpropagation to adaptive learning algorithms,” Computer Standards and Interfaces, vol. 16, Issue 3, pp. 265–278, 1994.

[97] CARUANA, R., and NICULESCU-MIZIL, A., “An empirical comparison of supervised learning algorithms,” ACM International Conference Proceeding Series, Vol. 148, Pittsburgh, PA, 2006.

[98] APPLEGATE, D. L., BIXBY, R. E., VASEK, C., and COOK, W. J., *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

[99] GUTIN, G., and PUNNEN, A. P. (Eds.), *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.

[100] BELLMAN, R., "Dynamic programming treatment of the traveling salesman problem," Journal of the ACM, vol. 9, Issue 1, pp. 61–63, 1962.

[101] WISKOTT, L., and SEJNOWSKI, T., "Slow feature analysis: Unsupervised learning of invariances," Neural Computation, vol. 14, Issue 4, pp. 715–770, 2002.

[102] NEYMAN, J., and PEARSON, E. S., "On the use and interpretation of certain test criteria for purposes of statistical inference: Part I," Biometrika, No. 20A(1/2), pp. 175–240, 1928.

[103] MOORE, J. E., and MOHR, C. F., "Biologically false positive seriologic tests for syphilis," Journal of the American Medical Association, vol. 150, Issue 5, pp. 467–473, 1952.

[104] PEPE, M. S., *The Statistical Evaluation of Medical Tests for Classification and Prediction*. Oxford University Press, 2003.

[105] BRATKO, A., FILIPIC, B., CORMACK, G. V., LYNAM, T. R., and ZUPAN, B., "Spam filtering using statistical data compression models," The Journal of Machine Learning Research, vol. 7, pp. 2673–2698, 2006.

[106] DUDA, R. O., and HART, P. E., *Pattern Classification*. Wiley, 2001.

[107] MURTAGH, F., "A survey of recent advances in hierarchical clustering algorithms," The Computer Journal, vol. 26, no. 4, pp. 354–359, 1983.

[108] DAY, W. H. E., and EDELSBRUNNER, H., "Efficient algorithms for agglomerative hierarchical clustering methods," Journal of Classification, vol. 1, no. 1, pp. 7–24, 1984.

[109] JOHNSON, S. C., "Hierarchical clustering schemes," Psychometrika, vol. 32, no. 3, pp. 241–254, 1967.

[110] Anon., "Getting Started with NuPIC", http://www.numenta.com/for-developers/software/pdf/nupic_gettingstarted.pdf (Accessed February 23, 2010).

[111] LEES, L., "Laminar heat transfer over blunt-nosed bodies at hypersonic speeds," Jet Propulsion, vol. 26, no. 4, pp. 259–269, 1956.

[112] FAY, J. A., and RIDDELL, F. R., "Theory of stagnation point heat transfer in dissociated air," Journal of Aeronautical Sciences, vol. 25, no. 2, pp. 73–85, 1958.

[113] ZOBY, E., and SULLIVAN, E., "Effects of corner radius on stagnation point velocity gradients on blunt axisymmetric bodies," Journal of Spacecraft and Rockets, vol. 3, no. 10, pp. 1567–1576, 1966.

[114] ANDERSON JR., J. D., *Hypersonic and High Temperature Gas Dynamics*. American Institute of Aeronautics and Astronautics, 1989.

[115] MATTINGLY, J. D., HEISER, W. H., and PRATT, D. L., *Aircraft Engine Design*. American Institute of Aeronautics and Astronautics, 2002.

[116] ANDERSON, J. D., *Fundamentals of Aerodynamics* (4th Ed.). McGraw Hill, 2007.

[117] JOHN, E. A. J., *Gas Dynamics* (2nd Ed.). Prentice Hall, 1984.

[118] VINCENTI, W. G., and KRUGER JR., C. H., *Introduction to Physical Gas Dynamics*. Krieger Publishing, 1965.

[119] Anon., Zeta1 Algorithms Reference (Version 1.3). Numenta Inc., 2007.

[120] BERKELEY, G., *A Treatise Concerning the Principles of Human Knowledge*. T. McCormack (Ed.), Open Court Publishing, 1904 (Reprint).

[121] IRANI, M., and PELEG, S., "Motion analysis for image enhancement: Resolution, occlusion, and transparency," Journal of Visual Communication and Image Representation, vol. 4, pp. 324–335, 1993.

[122] HOTTA, K., "Robust face recognition under partial occlusion based on support vector machine with local Gaussian summation kernel," Image and Vision Computing, vol. 26, Issue 11, pp. 1490–1498, 2008.

[123] GALVIN, B., MCCANE, B., and NOVINS, K., "Virtual snakes for occlusion analysis," *In Computer Vision and Pattern Recognition*. IEEE Computer Society, 1999.

[124] FORRESTER, J. W., *Road Maps: A Guide to Learning System Dynamics*. 1994.

[125] UNITED STATES DEPARTMENT OF DEFENSE, "Joint urban operations: Joint integrating concept", http://www.dtic.mil/futurejointwarfare/concepts/juo_jic_v1.pdf (Accessed February 22, 2010).

[126] Anon., "H.R. 1268: Emergency supplemental appropriations act for defense, the global war on terror, and tsunami relief", http://www.govtrack.us/congress/bill.xpd?bill=h109-1268 (Accessed February 22, 2010).

[127] MATHA, K., "A variable structure learning algorithm for multilayer perceptrons," *Intelligent Engineering Systems Through Artificial Neural Networks*. C. H. Dagli et al. (Eds.), ASME Press, 2004.

[128] MAKARUK, H. E., AUBREY, J. B., and HOLTKAMP, D. B., "Shock physics data reconstruction using support vector regression," International Journal of Modern Physics C, vol. 17, no. 9, pp. 1313–1325, 2006.

[129] Anon., Vitamin D Toolkit, version 1.3.0, http://www.vitamindinc.com/toolkit.html (Accessed February 22, 2010).

[130] UNITED STATES DEPARTMENT OF DEFENSE, "Report to Congress: Measuring stability and security in Iraq", September 2009, http://www.defense.gov/pubs/pdfs/9010_Report_to_Congress_Nov_09.pdf (Accessed February 23, 2010).